Calhoun: The NPS Institutional Archive

DSpace Repository

Theses and Dissertations 1. Thesis and Dissertation Collection, all items

2016-06

Modeling storm surges using discontinuous Galerkin methods

Hood, Karoline

Monterey, California: Naval Postgraduate School

http://hdl.handle.net/10945/49490

This publication is a work of the U.S. Government as defined in Title 17, United States Code, Section 101. Copyright protection is not available for this work in the United States.

Downloaded from NPS Archive: Calhoun

Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

http://www.nps.edu/library

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

MODELING STORM SURGES USING DISCONTINUOUS GALERKIN METHODS

by

Karoline Hood

June 2016

Thesis Advisor:	Frank Giraldo
Second Reader:	Simone Marras

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704–0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202–4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

1. AGENCY USE ONLY *(Leave Blank)*	**2. REPORT DATE** 06-17-2016	**3. REPORT TYPE AND DATES COVERED** Master's Thesis 07 July 14 to 17 June 16

4. TITLE AND SUBTITLE

MODELING STORM SURGES USING DISCONTINUOUS GALERKIN METHODS

5. FUNDING NUMBERS

6. AUTHOR(S)

Karoline Hood

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Naval Postgraduate School
Monterey, CA 93943

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

N/A

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

The views expressed in this document are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol Number: N/A.

12a. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

Storm surges have a devastating impact on coastlines throughout the United States. In order to accurately understand the impacts of storm surges there needs to be an effective model. One of the governing systems of equations used to model storm surges' effects is the Shallow Water Equations (SWE). In this thesis, we solve the SWE numerically by means of a discontinuous Galerkin (DG) method. The DG method provides high-order accuracy and geometric flexibility on unstructured grids. To run the model, we used both implicit and explicit time integration for solving the SWE. Using explicit time integration as our fundamental truth, we found the error norm of the implicit method to be minimal. This study focuses on the impacts of a simulated storm surge in La Push, Washington, which had undergone a beach restoration project. The beach restoration involved altering the bathymetry along the shoreline to prevent overtopping waves from breaching the mainland. To validate the simulations, we ran three benchmark tests. Real bathymetry was used along with real storm and tidal data. We measured the momentum flux of a wave on the existing bathymetry and the new bathymetry to determine if the new bathymetry had less momentum flux. Our results showed there was less momentum flux with the new bathymetry, and therefore the new bathymetry was more resistant to storm surges. After running the model at a high resolution, we modified the grid resolution to vary throughout the domain with a focus on high resolution closer to the shoreline. In our simulation, we also learned of the effects spurious waves can have on the results. Due to boundary conditions, a spurious wave can reflect back into a model and impact the velocity and momentum flux.

14. SUBJECT TERMS

Storm surges, Discontinuous Galerkin method (DG), GMSH, Shallow Water Equations (SWE), implicit and explicit time integrators, momentum flux, beach restoration, spurious waves

15. NUMBER OF PAGES 91

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

MODELING STORM SURGES USING DISCONTINUOUS GALERKIN METHODS

Karoline Hood
Captain, United States Army
B.S., United States Military Academy, 2006
M.S., University of Missouri Science and Technology, 2011

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN APPLIED MATHEMATICS

from the

NAVAL POSTGRADUATE SCHOOL
June 2016

Approved by: Frank Giraldo
Thesis Advisor

Simone Marras
Second Reader

Craig Rasmussen
Chair, Department of Mathematics

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Storm surges have a devastating impact on coastlines throughout the United States. In order to accurately understand the impacts of storm surges there needs to be an effective model. One of the governing systems of equations used to model storm surges' effects is the Shallow Water Equations (SWE). In this thesis, we solve the SWE numerically by means of a discontinuous Galerkin (DG) method. The DG method provides high-order accuracy and geometric flexibility on unstructured grids. To run the model, we used both implicit and explicit time integration for solving the SWE. Using explicit time integration as our fundamental truth, we found the error norm of the implicit method to be minimal. This study focuses on the impacts of a simulated storm surge in La Push, Washington, which had undergone a beach restoration project. The beach restoration involved altering the bathymetry along the shoreline to prevent overtopping waves from breaching the mainland. To validate the simulations, we ran three benchmark tests. Real bathymetry was used along with real storm and tidal data. We measured the momentum flux of a wave on the existing bathymetry and the new bathymetry to determine if the new bathymetry had less momentum flux. Our results showed there was less momentum flux with the new bathymetry, and therefore the new bathymetry was more resistant to storm surges. After running the model at a high resolution, we modified the grid resolution to vary throughout the domain with a focus on high resolution closer to the shoreline. In our simulation, we also learned of the effects spurious waves can have on the results. Due to boundary conditions, a spurious wave can reflect back into a model and impact the velocity and momentum flux.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

1 Introduction 1
1.1 Equations to Model Storm Surges 1
1.2 Solving the Shallow Water Equation 3
1.3 Storm Surges in La Push, Washington 4

2 Methodology 7
2.1 The Shallow Water Equations 8
2.2 Defining the Domain and Test Function 10
2.3 Integral Form 11
2.4 Semi-Discrete Equations 11
2.5 Matrix Form 13
2.6 Wetting and Drying 15
2.7 Time Integrators with Implicit and Explicit Methods 17

3 Verification and Test Cases 23
3.1 Case 1: Linear Bathymetry 25
3.2 Case 2: Linear Bathymetry with Shelf 27
3.3 Case 3: Linear Bathymetry with Tidal Pool 29

4 Simulation of A Storm Surge in La Push, Washington 33
4.1 Project Description 33
4.2 Metric Used to Measure Effectiveness of Bathymetry 44
4.3 Results From The Model 44
4.4 Error in Time Integration 48
4.5 Grid Refinement 51
4.6 Refined Boundary Conditions 59
4.7 Chapter Summary 60

5 Conclusion 61
5.1 Limitations 62

5.2 Future Work 62

List of References **65**

Initial Distribution List **69**

List of Figures

Figure 2.1 The total height of a fluid, where $H = h_S + h_B$. 9

Figure 3.1 Case 1: Testing simulation in 1D with linear bathymetry using fully explicit time integration. 26

Figure 3.2 Case 1: Testing simulation in 1D with linear bathymetry using fully implicit time integration. 26

Figure 3.3 Case 2: Testing simulation in 1D with changing sloping bathymetry to represent a shelf using fully explicit time integration. 28

Figure 3.4 Case 2: Testing simulation in 1D with changing sloping bathymetry to represent a shelf using fully implicit time integration. 29

Figure 3.5 Case 3: Testing simulation in 1D with changing sloping bathymetry to represent a tidal pool using fully explicit time integration. . . . 30

Figure 3.6 Case 3: Testing simulation in 1D with changing sloping bathymetry to represent a tidal pool using fully implicit time integration. . . . 31

Figure 4.1 Location of beach restoration in La Push, Washington. 34

Figure 4.2 Cross-section of existing and future bathymetry of La Push, Washington 35

Figure 4.3 Reproduction of old and new bathymetry 36

Figure 4.4 Location of Station 46041 with respect to La Push, Washington . 38

Figure 4.5 Location of proposed and actual location of field measurements conducted by USACE for La Push, Washington 38

Figure 4.6 Maximum wave heights at Station 46041 in 2014 39

Figure 4.7 Period between maximum waves at Station 46041 in 201 40

Figure 4.8 Difference in bathymetry resolution using 7520 and 22560 points. 43

Figure 4.9 Total norm momentum flux for the entire computational domain (30,000 meters). 45

Figure 4.10 Surface height of the wave prior to reaching the shoreline for old and new bathymetries during high-tide. 46

Figure 4.11 Momentum flux of the old and new bathymetries along the closest 100 meters to shoreline during high-tide. 46

Figure 4.12 Surface height of the wave prior to reaching the shoreline for old and new bathymetry during low-tide. 47

Figure 4.13 Momentum flux of different bathymetries along the shoreline during low-tide. 48

Figure 4.14 Grid divided into two planes: shoreline and deep ocean. 51

Figure 4.15 Sample of the deep ocean plane with 10 meter resolution using GMSH. 52

Figure 4.16 Near shore plane with resolution ranging from 1-10 meters using GMSH. 53

Figure 4.17 Comparison of Original grid and GMSH grid. 54

Figure 4.18 Momentum flux comparison during high tide using original and GMSH grid. 55

Figure 4.19 Momentum flux comparison during low tide using original and GMSH grid. 56

Figure 4.20 Comparison of old grid and GMSH grid. 57

Figure 4.21 Authentic wave and spurious wave. 58

List of Tables

Table 3.1 Surface height error norm for linear bathymetry 27

Table 3.2 Velocity error norm for linear bathymetry 27

Table 3.3 Surface height error norm for linear bathymetry with a shelf . . . 28

Table 3.4 Velocity error norm for linear bathymetry with a shelf 28

Table 3.5 Surface height error norm for linear bathymetry with a tidal pool . 31

Table 3.6 Velocity error norm for linear bathymetry with a tidal pool 31

Table 4.1 Norm error for the surface height of old bathymetry during high tide from 0 to 21 minutes 49

Table 4.2 Error norm for the velocity of old bathymetry during high tide from 0 to 21 minutes 49

Table 4.3 Error norm for the surface height of old bathymetry during low tide from 0 to 21 minutes 49

Table 4.4 Error norm for the velocity of old bathymetry during low tide from 0 to 21 minutes 49

Table 4.5 Error norm for the surface height of new bathymetry during high tide from 0 to 21 minutes 50

Table 4.6 Error norm for the velocity of new bathymetry during high tide from 0 to 21 minutes 50

Table 4.7 Error norm for the surface height of new bathymetry during low tide from 0 to 21 minutes 50

Table 4.8 Error norm for the velocity of new bathymetry during low tide from 0 to 21 minutes 50

THIS PAGE INTENTIONALLY LEFT BLANK

List of Acronyms and Abbreviations

AMR	Adaptive Mesh Refinement
BE	Backward Euler
CG	Continuous Galerkin Method
CFL	Courant-Friedrichs-Lewy Condition
DG	Discontinuous Galerkin Method
FD	Finite-Difference Method
FE	Forward Euler
GMRES	Generalized Minimal Residuals
JFNK	Jacobian-free Newton-Krylov
LGL	Legendre-Gauss-Lobatto
MOL	Method of Lines
NOAA	National Oceanic and Atmospheric Administration
NPS	Naval Postgraduate School
NRPC	Non-reflecting Boundary Condition
ODE	Ordinary Differential Equation
PDE	Partial Differential Equation
PL	Public Law
RAM	Random Access Memory
RK4	Runge-Katta method of order 4
SWE	Shallow Water Equations

USACE	United States Army Corps of Engineers
USCG	United States Coast Guard

Executive Summary

From flooding to erosion, storm surges have a devastating impact on coastlines throughout the United States. To accurately predict storm surges and understand their impacts, you need to have an effective model to simulate their destructive capabilities. These simulations are often done through numerical methods that solve the Shallow Water Equations (SWE).

The SWE are a system of time dependent, hyperbolic, nonlinear, partial differential equations (PDE) that are used when the fluid is incompressible and the depth is shallow [1]. Historically, the main challenge in solving the SWE involves balancing the accuracy of the model with computational efficiency. This challenge has prompted numerical methods to evolve. Although not widely used, the Discontinuous Galerkin (DG) method is growing in popularity [2]. Our model uses the DG method on quadrilaterals with both implicit and explicit time integrators.

Prior to running realistic simulations, it is critical to verify the method you are using to solve that model. For our model, we used three test cases of the DG method with a nodal basis on quadrilaterals for verification. Since no experimental solution was feasible, results from the models were compared to those of other numerical simulations. During verification of our method, we also determined the error norm for the implicit method compared against the explicit method, which we considered the fundamental truth. Proving the effectiveness of the implicit time integrator gave us the flexibility to use large time-steps for future work.

After validating our method, we used a real-world case study from a United States Corps of Engineers (USACE) beach restoration project to determine the changes in momentum flux from a wave due to altering the bathymetry. Our real-world case study compared the momentum flux for two different types of bathymetry based on field data from La Push, Washington. We also analyzed two different tide levels for each bathymetry. We used explicit and implicit time integrators to model the two different tides.

Our model required many inputs. Our new and old bathymetry were replicated from cross-sectional data provided by USACE. To determine the magnitude of our storm surge, we obtained both field data from USACE and also data provided by the National Oceanic and Atmospheric Administration (NOAA). Using this data, we then determined our storm surge to be located 30,000 meters away from La Push, Washington and to have a base of 20 meters with an amplitude of 11 meters for a worst-case scenario [3] and [4]. Additionally, we wanted to run our model at both high and low tide. Based on annual field data from U.S. Harbors, La Push has an extreme high tide of 2.96 meters and an extreme low tide of 0.67 meters [5]. We also included a Manning's Roughness Coefficient of 0.02 to replicate the friction from the sand interacting with the bottom of the ocean floor [6].

Given our large domain (30,000 meters), the computational cost to run the model would be high. While we wanted an accurate and unique solution, we also wanted to have a computationally efficient model. Therefore, to reduce the computational cost, we modified the grid resolution. To maintain high-order accuracy, we used a fourth order polynomial, two elements in the y-direction, and 5640 elements in the x-direction, which gave a total of 22560 points.

Lastly, we decided our model should run with a time restart of every 45 seconds (0.75 minutes) with a time-step of 0.05 both implicitly and explicitly for a total of 2100 seconds (35 minutes). We kept the same time-step to compare the difference in error between implicit and explicit integration. We wanted to see how different the implicit solution would be from the explicit. If we chose a different time-step, then we would introduce errors of increased complexity in the time-integrator accuracy and would not be able to compare implicit with explicit. We determined that fifteen minutes was the approximate time the wave would take to travel 30,000 meters and interact with the bathymetry.

Our results showed that the wave had less momentum flux with the new bathymetry for both high-tide and low-tide conditions, and therefore, this bathymetry would be more effective in mitigating the destruction of storm surges. For high tide, the peak momentum flux for the old bathymetry was $3.9\frac{m^3}{s^2}$ compared to the new bathymetry, which had a peak of $2.9\frac{m^3}{s^2}$ or a decrease of 1.0 $\frac{m^3}{s^2}$. For low tide, the old bathymetry had a maximum momentum flux of $1.12\frac{m^3}{s^2}$ compared to the new bathymetry, which had a maximum momentum flux of $0.41\frac{m^3}{s^2}$ or a decrease of 0.79 $\frac{m^3}{s^2}$. The scale of variance from the low tide compared to the high tide means that the new bathymetry would be more effective during a storm surge in low tide conditions compared to high tide conditions. We computed the 1-Norm, 2-Norm, and the ∞-Norm of the surface height and velocity, for both low tide and high tide. Our results showed that the error norm for implicit integration compared to explicit integration was minimal and ranged in magnitude from 10^{-2} to 10^{-1}.

After running the model at a high resolution, we modified the grid resolution using a program called GMSH to vary throughout the domain with a focus on high resolution closer to the shoreline where we were more concerned [7]. While some information was lost using a coarser grid, the modification allowed us to save in computational time by a factor of 6.7.

For future work, we would establish a non-reflecting boundary condition (NRBC) to eliminate a spurious wave in our model from hitting the right boundary wall and reflecting towards the shoreline. This was an issue we identified when comparing the GMSH grid to the original grid. The spurious wave had an impact on our velocity and momentum flux values, which was magnified on the GMSH grid.

In addition, the next logical step would incorporate multiple waves into the model. Multiple waves would add an extra layer of complexity by incorporating the energy caused by waves interacting with each other. In addition, all our simulations were in 2-D but for a 1-D initial condition. In the future, we could adapt the simulations to 3-D. Since SWE are 2-D, to run a full 3-D ocean simulation you need to switch to a Navier-Stokes solver. Moreover, the combination of 3-D with adaptive mesh refinement (AMR) would provide a computationally efficient model that could focus on the changes in bathymetry of the last 100 meters in our domain [8].

Another area for future consideration is the movement of sediment due to erosion from storm surges. Modeling sediment movement has been a difficult problem for researchers studying coastal communities and navigation channels due to the complexity of the problem. The Advection-Diffusion Equation is a PDE used to numerically represent sediment movement, and the DG method could be used to solve this PDE [9].

References

[1] B. Bonev, "Discontinuous Galerkin methods for shallow water equations," Master's thesis, Department of Computational Mathematics and Simulation Science, EPFL Lausanne, Lausanne, Switzerland, November 2015.

[2] S. Marras, J. Kelly, M. Moragues, A. Meller, M. Kopera, M. Vazquez, F.X. Giraldo, G. Houzeaux, and O. Jorba, "A review of elements-based Galerkin methods for numerical

weather prediction. finite elements, spectral elements, and discontinuous Galerkin," *Archives of Computational Methods in Engineering*, 2015.

[3] N. Oceanic and A. A. (NOAA). (2015, September). National Data Buoy Center. [Online]. Available: http://www.ndbc.noaa.gov/station_history.php?station=46041

[4] I. Evans-Hamilton, "FY 2010 field data collection program at the Quillayute navigation project, La Push, WA: Data report," Evans-Hamilton, INC., Tech. Rep., June 2010.

[5] U. Harbors. (2015). Washington tide charts US harbors. [Online]. Available: http://wa.usharbors.com/monthly-tides/Washington-Columbia

[6] G. J. Arcement, "Guide for selecting Manning's roughness coefficients for natural channels and flood plains," United States Geological Survey, Tech. Rep., 1989.

[7] C. Geuzaine and J.-F. Remacle, "GMSH: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities." *International Journal for Numerical Methods in Engineering*, vol. 79, no. 11, pp. 1309–1331, 2009.

[8] M. A. Kopera and F. X. Giraldo, "Analysis of adaptive mesh refinement for IMEX discontinuous Galerkin solutions of the compressible Euler equations with applications to atmospheric simulations," *Journal of Computational Physics*, vol. 271, pp. 92–117, 2014.

[9] P. Y. Julien, *Erosion and Sedimentation*. Cambridge, England: Cambridge University Press, 2010.

Acknowledgments

I would like to first thank my thesis advisor, Prof. Frank Giraldo, for pushing me well beyond my comfort zone in mathematics and computer science. Without his mentorship and dedication, this thesis would not have been possible. I am especially grateful that he offered his free time to help me resolve and understand both physical and theoretical questions in applied mathematics. Also, I am grateful to my second reader, Simone Marras, for the feedback he gave me, which provided a different perspective to modeling storm surges. I would also like to thank Michal Kopera, who helped me debug my code and process data on Hamming.

I am thankful to the USACE Seattle District for the field data on La Push, Washington, and the inspiration for my thesis. I am always humbled and honored to be able to have the opportunity to serve with the U.S. Army Corps of Engineers. Essayons!

Also, I am thankful to the NPS Applied Mathematics Department. Without the department, I would not have had the skills and foundation to write my thesis. A special thanks goes to Bard Mansager for helping me get into NPS and preparing me for my follow-on assignment to teach at the United States Military Academy. To my comrades Scott Warnke and Ryan Miller, thank you for the many late evenings and weekends discussing math, MATLAB and Python. I am excited to know we will be instructors together at West Point.

I would also like to thank LT Tucker Freismuth for assisting me with MATLAB and providing a plethora of knowledge on oceanography. In addition, I would like to thank Michelle Pagnani from the NPS Graduate Writing Center and Mark Gondree from the Computer Science Department.

As always, I would like to thank my parents, Dan and Gisela Hood. They taught me at a young age that the sky is the limit and to persevere whenever I fall. To my brother, Daniel Hood, I would like to thank you for keeping me grounded.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1:
Introduction

From flooding to erosion, storm surges have a devastating impact on coastlines throughout the United States. Along both the west and east coast in the United States, there are large death tolls after storm surges. For example, 1,500 people died from the storm surges produced by Hurricane Katrina (in 2005) [1]. To accurately predict storm surges and understand their effects, you need to have an effective model to simulate the destructive capability. These simulations are often done through numerical methods that solve shallow water equations (SWE). SWE were one of the first applications for using numerical methods with digital computers that were available in the 1940s [2]. Computational efficiency and numerical accuracy represent two important challenges in the solution of SWE with a computer. These challenges have prompted numerical methods to evolve. Although not widely used due to its high-order accuracy and low dispersion error, the discontinuous Galerkin (DG) method is gaining momentum to solve these problems and is slowly replacing the more classical numerical approximation by finite volumes [3].

1.1 Equations to Model Storm Surges

The non-linear SWE are extremely useful in describing motions of fluids for the deep ocean, coastal ocean, estuaries, rivers, open channels, and coastal floodplains with irregular coastal boundaries [4]. In particular, they are ideal for modeling storm surges where the reliability of the models depends highly on the quality of the input data [2]. The SWE are a system of time-dependent hyperbolic nonlinear partial differential equations (PDE) that are used when the fluid is incompressible and the depth is shallow relative to the length of the wavelength [5]. The SWE have a hyperbolic continuity equation to account for water elevation and are coupled with a momentum equation to account for the average depth velocity [6]. One of the main assumptions of the SWE is that the depth of the water is significantly smaller than the length of the wave, making this equation appropriate for storm surges. In addition, the assumption is made that there is little variation in the stratification of the fluid (less than a few percent) due to uniform temperature and salinity. Moreover, the fluid is considered independent of pressure and therefore incompressible [2].

According to [7], in vector form the SWE of two-dimensional flows can be expressed as

$$\frac{\partial \boldsymbol{q}}{\partial t} + \boldsymbol{\nabla} \cdot \boldsymbol{F}(\boldsymbol{q}) = S(\boldsymbol{q}), \tag{1.1}$$

where $\boldsymbol{q} = (\phi, \phi \mathbf{u}^T)^T$ represents the conservation variables and T is the transpose of the solution vector [8].

The flux tensor is

$$\mathbf{F}(\mathbf{q}) = \left(\begin{array}{c} \phi \mathbf{u} \\ \phi \mathbf{u} \otimes \mathbf{u} + \frac{1}{2}\phi^2 \mathcal{I}_{\mathbf{2}} - \mu \nabla(\phi \mathbf{u}). \end{array} \right) \tag{1.2}$$

The source function is

$$S(\mathbf{q}) = -\left(\begin{array}{c} 0 \\ f(\mathbf{k} \times \phi \mathbf{u}) + \phi \nabla \phi^b - \frac{\tau}{\rho} + \gamma \phi \mathbf{u}, \end{array} \right) \tag{1.3}$$

where $\boldsymbol{\nabla} = (\partial_x, \partial_y)^T$,

$\otimes$ is the tensor product operator,

$\phi = gh$ is the geopotential height,

h is the total height of the surface height of the fluid,

g is the magnitude of acceleration due to gravity,

ϕ^b is the distance from the floor to the bathymetry,

$\mathbf{u} = (u, v)^T$ is the velocity vector,

$f = f_0 + \beta(y - y_m)$ is the Coriolis parameter,

$k = (0, 0, 1)^T$ is the normal vector of the x-y plane,

$\boldsymbol{\tau}$ is the wind stress,

and γ is the bottom friction.

$\mathbf{I}_2$ is the rank-2 identity matrix.

Unless otherwise stated, it can be assumed that the International System of Units (SI) is being used; units of length are in meters, and time is measured in seconds.

1.2 Solving the Shallow Water Equation

In general, there is no analytical way to solve the non-linear SWE and therefore numerical methods are required. To solve this system of partial differential equations (PDE), discretization of physical space is required. Following this approach, space is partitioned into elements in order to approximate the solutions to the equations.[1] Historically, SWE have been solved with the finite-difference (FD) method, which uses a uniform grid. Advantages of the FD are that it is simple to implement and overall relatively straightforward [3]. One of the major restrictions on FD is that it has limited geometric flexibility. To overcome the restriction, the continuous Galerkin (CG) was introduced to allow the use of unstructured grids for geometric flexibility [7].

In 1997, high-order continuous Galerkin methods were used for the SWE on a sphere [9]. In 2002, Giraldo et al. [10] introduced an efficient DG method for the SWE. One of the main advantages of the DG method is that it is an extremely robust approach that was designed explicitly for hyperbolic PDEs (wave-like phenomena) [10].

The DG method has high-order accuracy, and flexibility in unstructured grids [11]. Another significant advantage of DG is its conservation properties. While the CG method has global conservation properties, the DG method is both locally and globally conservative [12]. It can be described as a combination of the finite volume method and the finite element method with high-order accuracy [5]. It was first used for the planar SWE by Schwanenberg and Kongeter [7].

One of the initial drawbacks of the DG method as compared to the CG method was its large computational cost. The DG method's large number of degrees freedom allowed for enhanced capability but also lead to a higher computational cost. Studies showed that there was a four to five times greater computational cost using DG compared to CG for linear interpolation [11]. The high computational cost helped motivate the use of quadrilateral-based elements. Two adjacent triangle elements can be merged into a quadrilateral-based element; therefore, fewer elements are required [11]. In this thesis, the model uses quadrilateral elements.

---

[1] Analytic solutions exist only for very simple special cases.

To solve time dependent PDEs, like the SWE, you need to include time integrators. Traditionally, the most common time integrator used is the explicit method. The explicit method solves the SWE using information at the current time to compute the solution at the next forward time level [2]. Arguably, the most common explicit method is the Runge-Kutta method of order 4 (RK4). RK4 is an ideal explicit method because it has a local truncation error of $O(h^4)$ and eliminates the requirement to compute the derivative at the current time state. Truncation error is the measurement of accuracy from the expansion of the Taylor polynomial [13]. The other class of method, the implicit time integrator, uses both the current state and later state of the system [2]. Simply put, the implicit method requires the solution of the linear algebra problem $\mathbf{Ax} = \mathbf{b}$ where $\mathbf{A} \in R^{NxN}$ and $\mathbf{x}, \mathbf{b} \in R^N$. N is defined as the degrees of freedom. In the explicit method, $\mathbf{A}$ is a diagonal matrix and therefore is easier to solve. Advantages of the implicit method are that it allows infinitely large time-steps whereas the explicit has to satisfy the Courant-Friedrichs-Lewy (CFL) condition. The CFL condition is a restriction on the maximum allowable time-step one can use for solving the PDE. The restriction is in place to ensure stability [14]. Although a stable solution is obtained with infinitely large time steps using the implicit method, it is still important to have an appropriate time step size in order to obtain an accurate solution. This thesis will compare both the explicit and implicit time integrators used to solve the SWE, weighing the stability and accuracy of each integrator.

1.3 Storm Surges in La Push, Washington

This thesis will focus specifically on modeling the effects of storm surges in La Push, Washington. In La Push, seasonal storm surges require annual restoration projects by the United States Army Corps of Engineers (USACE). The town lies within the Quileute Indian tribal reservation, which is on the northwest coast of the Olympic Peninsula, and through the request of the Quileute Nation, the Corps of Engineers provides beach protection from these surges [15].

Field studies have been conducted in this area that show storm surges with maximum wave heights reaching approximately 11 meters and an average wave height of approximately 6 meters [16]. There are several ways to mitigate the impacts of these surges including the construction of levees/jetties or the repair of breached areas with locally dredged fill. In La Push, many different methods have been used to protect the area dating back to the 1950s. Currently, USACE is restoring an eroded beach with local fill with the intent to mitigate damage from storm surges and wave run-ups [15].

To understand the effectiveness of the beach restoration project in La Push, this thesis will use numerical simulations to model storm surges interacting with the existing bathymetry and the new bathymetry that has been filled with dredged material. The first step in modeling storm surges is to simplify the problem by using the non-linear SWE and use proven cases to verify the accuracy of numerical simulations prior to modeling the restoration project. Once verification is complete, further analysis on wave height and momentum flux can be done to determine how effective the beach restoration project will be for a future storm surge. The model will use the DG method on quadrilaterals using both implicit and explicit time integrators.

The remainder of this study is organized as follows. Chapter 2 describes the methodology used to solve the SWE equations. Chapter 3 describes the verification of the code via three test cases. Chapter 4 shows the numerical simulations of a storm surge in La Push, Washington, with its existing grade and a model of a storm surge with the newly constructed bathymetry using both implicit and explicit time integrators. The final chapter presents the conclusion and recommendation.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 2:
Methodology

There are many difficulties in solving the SWE numerically. As previously discussed, one solution to solving these equations is with the discontinuous Galerkin (DG) method. While the finite volume method is one of the most accepted methods for solving geophysical fluid dynamics for ocean and shallow water models, the DG method is slowly gaining momentum. For example, Delft-3D uses a form of finite elements and finite volumes and is one of the most common types of software used by engineers (including USACE) to model hydrodynamics and sediment transport [17]. The DG method is considered expensive compared to the finite volume method due to redundant degrees of freedom. Although expensive, the DG method is considered a successful strategy because it provides high-order relative to low-order finite volume methods, it is robust, and it is flexible for unstructured grids. The unstructured grids allow for more accurate representation of coastlines and simulations of wave processes [7].

As discussed in Chapter 1, the quadrilateral-based DG method is considered by researchers to be superior to the triangular method with respect to cost per accuracy and overall computing time. One of the largest computational costs in DG methods is the evaluation of area integrals and edge integrals. With the use of the quadrilateral method, computational efficiency improves significantly. The reduction in cost is done by merging two adjacent elements. A mesh of quadrilateral-based elements has approximately two-thirds as many edges and half as many elements [11]. The efficiency of quadrilaterals over triangles is that on quadrilaterals, the Legendre-Gauss-Lobatto (LGL) nodes can be used for both interpolation and integration. On triangles, you have to separate the set of points (one for interpolation, the Fekete points, another for integration, Gauss points). In other words, on the triangle there is no set of points used for both interpolation and integration. The quadrilateral-based approach allows for a simpler algorithm [11].

Lastly, one of the main challenges in solving the SWE or any time dependent PDE is the type of time integrator used. Time integrators are required in order to advance the solution in time while maintaining higher order accuracy and stability [7]. Explicit and implicit time integrators may be used for advancing the solution.

2.1 The Shallow Water Equations

As defined by Giraldo [18], in conservation 1-D form the shallow water equations are written as

$$\frac{\partial h_s}{\partial t} + \frac{\partial}{\partial x}(U) = 0 \tag{2.1}$$

and

$$\frac{\partial U}{\partial t} + \frac{\partial}{\partial x}\left(\frac{U^2}{h} + \frac{1}{2}gh_S^2\right) = -gh_b\frac{\partial h_s}{\partial x}, \tag{2.2}$$

where

U = hu,

g acceleration due to gravity ($9.8\frac{m}{s^2}$),

u is the velocity ($\frac{m}{s}$), and

h is the total height of the surface height of the fluid. The surface height is the sum of h_S (surface height from the mean level) and h_B (distance from mean level to the basin) [18]. See Figure 2.1.

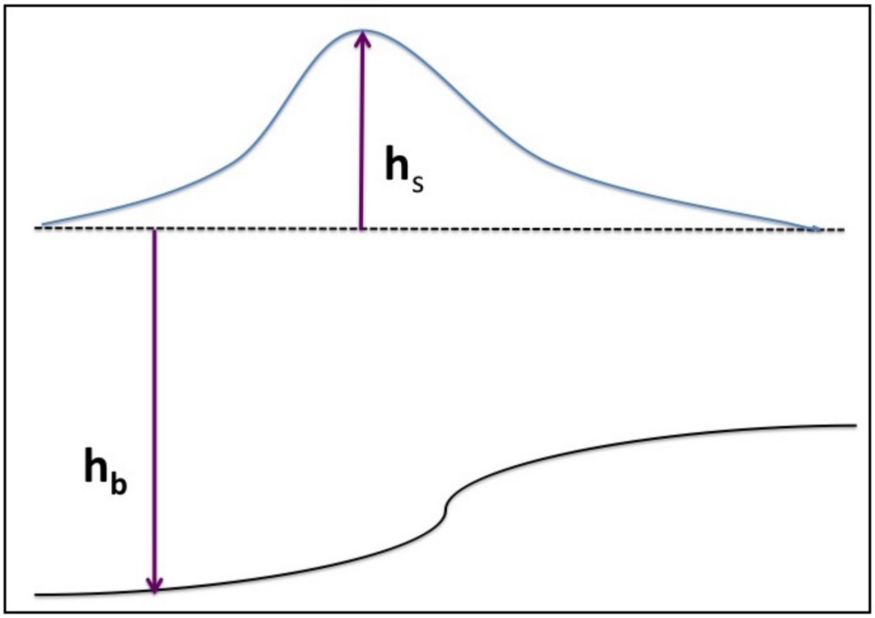


Figure 2.1: The total height of a fluid, where $H = h_S + h_B$. Adapted from [18].

Giraldo et al. [18] defines the SWE in a more compact form as

$$\frac{\partial q}{\partial t} + \frac{\partial F(q)}{\partial x} = \mathbf{S}(\mathbf{q}), \tag{2.3}$$

where $\mathbf{q}$ is the vector of the conservation variables, $\mathbf{q} = \begin{pmatrix} h_S \\ U \end{pmatrix}$;

$\mathbf{F}(\mathbf{q})$ is the flux tensor, $\mathbf{F}(\boldsymbol{q}) = \begin{pmatrix} U \\ \frac{U^2}{h} + \frac{1}{2} g h_S^2 \end{pmatrix}$; and

$\mathbf{S}(\mathbf{q})$ is the source function, $\mathbf{S}(\boldsymbol{q}) = -\begin{pmatrix} 0 \\ -h_B \frac{\partial h_s}{\partial x} \end{pmatrix}$ [18].

As previously defined in Chapter 1 [18], from 1-D to 2-D the SWE in compact differential form is written as

$$\frac{\partial \boldsymbol{q}}{\partial t} + \boldsymbol{\nabla} \cdot \boldsymbol{F}(\boldsymbol{q}) = S(\boldsymbol{q}). \tag{2.4}$$

From the compact differential form, we must transform the equations into integral form, but first we must define the domain and test function we will be using.

2.2 Defining the Domain and Test Function

The first step to solving the SWE using the DG method is to define discrete operators that will be used for approximating the solution. Following Giraldo [19], Ω is considered the domain we integrate over, and Γ defines the boundaries of the integral [14]. Since we numerically integrate all functions, we must computationally decompose the domain, Ω, into N_e non-overlapping quadrilateral elements Ω_e such that

$$\Omega = \bigcup_{e=1}^{N_e} \Omega_e. \tag{2.5}$$

To decompose the domain into quadrilaterals, we transform from physical Cartesian coordinates $\mathbf{x} = (x, y)^T$ to a computational space, also known as a local coordinate system, $\boldsymbol{\xi} = (\xi, n)^T$ [7].

In DG, a basis function is used to approximate the variables within each element Ω_e. With the use of quadrilaterals, the fastest and most efficient way to solve the PDE is with a tensor product 1-D basis. We can use the notation $\psi_i(\xi,\eta) = h_j(\xi) \otimes h_k(\eta)$, where $(\xi,\eta) \in [-1,+1]^2$.

The basis function in 1-D is h, and j, k ranges from 0 to N, to transform from 1-D to 2-D, i is mapped to $k(N+1)+j+1$ where i is the local index [7].

The local element solution, $\mathbf{q}$, is approximated with an N^{th} order polynomial as

$$\mathbf{q_N}(\xi) = \sum_{k=1}^{M_N} \psi_i(\boldsymbol{\xi}) \mathbf{q_n}(\boldsymbol{\xi_i}). \tag{2.6}$$

Since we are using quadrilaterals, $M_N = (N+1)^2$, where M_N is the number of interpolation points in Ω_e. By multiplying our test function and integrating the local domain, we will be able to obtain the weak integral form of the SWE [14].

2.3 Integral Form

With a given function f(x), we can write the integrals required in Galerkin methods with a tensor product of the 1-D integration formulas. The formula is then mapped from the physical to the computational space as follows:

$$\int_{x_0}^{x_1}\int_{y_0}^{y_1}\mathbf{f}(x,y)dxdy = \int_{-1}^{+1}\int_{-1}^{+1}\mathbf{f}(\xi,\eta) \mid \mathbf{J}(\xi,\eta) \mid d\xi d\eta, \tag{2.7}$$

where

$\mid \mathbf{J} \mid$ is the determinant of the Jacobian when you map from (x, y) to (ξ,η).

The integral is then rewritten as the Riemann sum

$$\int_{-1}^{+1}\int_{-1}^{+1} f(\xi,\eta) \mid J(\xi,\eta) \mid d\xi d\eta \approx \sum_{i=0}^{Q}\sum_{j=0}^{Q} w_i w_j f(\xi_i,\eta_j) \mid J(\xi_i,\eta_j) \mid, \tag{2.8}$$

where w_i and w_j are 1-D quadrature weights and points,

ξ and η are the coordinates in computational space [14], and

Q is the order quadrature approximation [7].

Since we are using quadrilaterals, inexact integration can be used, therefore fourth order (N=4) or higher polynomial is required in order to be able to integrate the function without incurring too much error [20]. The advantages of inexact integration are we can still obtain a stable solution without sacrificing computational cost. Typical polynomials used are Lagrange and Legendre polynomials to interpolate data with given points that are constructed from the tensor product of 1-D basis functions. In addition, Lagrange polynomials can be used for 2-D (such as on triangles) but are not tensor product based [13].

2.4 Semi-Discrete Equations

To solve the governing equations numerically, we must first derive a discrete representation of the PDE. A common method is to discretize the spatial operators and ignore the temporal derivatives first, changing your PDE to an ordinary differential equation (ODE) and then discretize the remaining system with respect to time [5]. This is known as the method of lines (MOL) where the spatial and temporal derivatives are handled separately [21].

Using the Ω_e non-overlapping cells, and by substituting $\mathbf{q}_N^{(e)}, \mathbf{u}_N^{(e)}$, and $\mathbf{f}_N^{(e)}$ into the compact SWE (Equation 2.7), we are able to obtain a weak form of the governing equations written as

$$\int_{\Omega_e} \psi(x) \left(\frac{\partial q_N}{\partial t} - \mathbf{F_N} \cdot \boldsymbol{\nabla} - S_N \right) d\Omega_e + \int_{\Gamma_e} \psi(x) \mathbf{n} \cdot \mathbf{F_n}^* d\Gamma = 0 \tag{2.9}$$

where $\mathbf{F_N} = \mathbf{F}(\mathbf{q_N})$, defined in Equation 1.2,

$\mathbf{S_N} = \mathbf{S}(\mathbf{q_N})$, defined in Equation 1.3,

$\mathbf{n}$ is the outward unit normal vector,

$\boldsymbol{\nabla} = (\frac{\partial}{\partial x}\hat{\imath} + \frac{\partial}{\partial y}\hat{\jmath})$ is the 2-D gradient operator, and $\hat{\imath}$ and $\hat{\jmath}$ are the directional unit vectors in 2-D Cartesian space [18],

Γ_e is the element edge [7], and

$\mathbf{F_N^*}$ is the numerical flux. A common numerical flux is the Rusanov flux. The Rusanov flux is the average of the flux of two elements sharing the same edges with the addition of $(\mid \lambda \mid)$. λ is the maximum wave speed of the system [22] and represents the maximum eigenvalue of the Jacobian Matrix [18]. λ is defined as

$$\lambda = max(\mid U^L \mid + \sqrt{\phi^L}, \mid U^R \mid + \sqrt{\phi^R}) \tag{2.10}$$

Therefore, the equation for the Rusanov flux [18] is

$$\mathbf{F_N^*} = \frac{1}{2}[\mathbf{F_N}(\mathbf{q_N^L}) + \mathbf{F_N}(\mathbf{q_N^R}) - \mid \lambda \mid \left(\mathbf{q_N^R} - \mathbf{q_N^L} \right) \mathbf{n}] \tag{2.11}$$

where $U^{L,R} = \mathbf{u^{l,r}} \cdot \mathbf{n}$ and is the normal component of the velocity for the edge Γ_e, L and R are the left and right side element edges, and

$\mathbf{n}$ is the normal outward pointing vector.

By applying a second integration by parts, we obtain the strong integral form

$$\int_{\Omega_e} \psi_i(x) \left(\frac{\partial q_N}{\partial t} + \boldsymbol{\nabla} \cdot \mathbf{F_N} - S_N \right) dx = \int_{\Gamma_e} \psi_i(\mathbf{x}) \mathbf{n} \cdot \left(\mathbf{F_N} - \mathbf{F_N^*} \right) dx \tag{2.12}$$

where $\mathbf{F_N^*}$ is the Rusanov flux [7].

2.5 Matrix Form

As mentioned earlier, we substitute the equation

$$\mathbf{q}_N = \sum_{i=1}^{M_N} \psi_i \mathbf{q}_i \tag{2.13}$$

into Equation 2.12, to form the semi-discrete representation of the PDE. Our new equation becomes

$$\int_{\Omega_e} \psi_i \psi_j dx \frac{dq_j}{dt} + \int_{\Omega_e} \psi_i \boldsymbol{\nabla} \psi_j dx \cdot \mathbf{F}_j - \int_{\Omega_e} \psi_i \psi_j dx S_j = \int_{\Gamma_e} \psi_i \psi_j \mathbf{n} dx \cdot (\mathbf{F} - \mathbf{F}^*)_j \tag{2.14}$$

Written in index notation, we define the elemental matrices as

$M^e_{ij} = \int_{\Omega_e} \psi_i \psi_j d\Omega$, $\mathbf{M}^S_{ij} = \int_{\Gamma_e} \psi_i \psi_j \mathbf{n} d\Gamma$, $\mathbf{D}^e_{ij} = \int_{\Omega_e} \psi_i \boldsymbol{\nabla} \psi_j d\Gamma$.

Using the elemental matrices, mass ($M^{(e)}_{ij}$), flux ($F^{(e)}_j$), and differentiation ($D^{(e)T}_{ij}$) we write the SWE as

$$M^{(e)}_{ij} \frac{dq^{(e)}_j}{dt} + (\mathbf{D}^{(e)T}_{ij}) F^{(e)}_j - M^e_{ij} S_j = (\mathbf{M}^s_{ij})^T (\mathbf{F} - \mathbf{F}^*)^e_j, \tag{2.15}$$

where e refers to an elemental object. In this case, the elemental object is the matrix.

The total time-derivative q is $\frac{dq_j}{dt}$.

The side base or edge-based evaluation of the elemental matrices is s [7].

When using inexact integration where interpolation and integration points can be co-located, the matrices simplify to

$M_{ij}^{(e)} = w_i \mid J_i \mid \delta_{ij}$,

$M_{ij}^{(l)} = w_i^{(l)} \mid J_i^{(l)} \mid \delta_{ij}$, and

$D_{ij}^{(e)} = w_i^{(e)} \mid J_i^{(e)} \mid \nabla\psi_j(x_i)$,

where δ is the Kronecker delta function, which is equal to 1 if $i = j$ and 0 if $i \neq j$ [8].

From our defined matrices and dividing by the mass matrix, the SWE become

$$\frac{dq_i^{(e)}}{dt} + (\boldsymbol{\nabla}\psi_j(x_i))^T \mathbf{F_j^{(e)}} = S_i^{(e)} + \sum_{l=1}^{4} \tau_i^{(l)} Q_i^{(l)} \mathbf{n_i^{(e,l)}} \cdot (\mathbf{F_i^{(e)}} - \mathbf{F_i^{(*,l)}}) \tag{2.16}$$

where $Q = 1$ if i is on the edge and 0 otherwise, and

$\tau_i^{(l)} = \frac{w_i^{(l)}|J_i^{(l)}|}{w_i^{(e)}|J_i^{(e)}|}$ [23].

To simplify τ, we assume $\xi = \xi(x)$ and $\eta = \eta(y)$, which allows for the computational axes to range with the physical axes. We can then write ξ and η as

$\xi = \frac{2(x-x_0)}{\Delta x} - 1$ and

$\eta = \frac{2(u-y_0)}{\Delta y} - 1$,

where x_0 and y_0 are the bottom corner points and Δx and Δy are the lengths of the direction of the elements [23].

The mapping from computational to physical axes then yields

$$\frac{\partial \xi}{\partial x} = \frac{2}{\Delta x} \frac{\partial \eta}{\partial y} = \frac{2}{\Delta y} \tag{2.17}$$

and

$$J^{(e)} \equiv \frac{\partial x}{\partial \xi} \frac{\partial y}{\partial \eta} - \frac{\partial x}{\partial \eta} = \frac{\Delta x \Delta y}{4}, \tag{2.18}$$

$\mid J^{(l)} \mid = \frac{\Delta y}{2}$ for the left-right edge and $\frac{\Delta x}{2}$ for the top-bottom edge and

$\tau_i^{(l)} = \frac{2}{w\Delta x}$ for the left-right edge and $\frac{2}{w\Delta y}$ for the top-bottom edge [23].

Using our previously defined numeral flux function (the Rusanov flux), the SWE is written as

$$\frac{dq_i^{(e)}}{dt} + (\nabla \psi_j(x_i))^T F_j^{(e)} = S_i^{(e)} + \sum_{l=1}^{4} \hat{\tau}_i^{(l)} Q_i^{(l)} \mathbf{n}_i^{(e,l)} \cdot [F_i^{(e)} - F_i^{(l)} - \delta \mid \lambda_{max} \mid \mathbf{n}_i^{(e,l)} (q_i^{l)} - q_i^{(e)})], \tag{2.19}$$

where $\hat{\tau} \equiv \frac{\tau}{2} = \frac{1}{w\Delta s}$ and

$\Delta s = (\Delta x, \Delta y)$ [23].

2.6 Wetting and Drying

When dealing with shallow water bodies, there is always the concern of boundary conditions. In particular, there is the concern of how to model wetting and drying as time advances in the model. One of the main issues with wetting and drying is the lack of knowledge concerning the resistance laws for a thin layer of water interacting with ground irregularities [24] and [25]. In addition, there is a numerical problem of ensuring stability and conservation of mass [26]. Issues of conservation of mass occur when there is negative water height in a model.

The DG method allows for the slope of the water height to be modified in order to allow positive water height. Y. Xing, X. Zhang, and C. Shu [27] were able to overcome the resistance laws of wetting and drying by introducing a simple positivity preserving limiter, which modified the water height slope. This limiter can be used in the DG method under suitable CFL conditions and allows for the water height to remain non-negative and also still preserve the mass conservation with no impact on high-order accuracy. Y. Xing, X. Zhang, and C. Shu [27] used several numerical examples to confirm the accuracy, stability, and conservation in mass using a limiter in the DG method.

We use the Y. Xing, X. Zhang, and C. Shu limiter in our model. The new solution variable is:

$$q_i^{new} = q_{mean} + \alpha \left(q_i^{(e)} - q_{mean} \right), \tag{2.20}$$

where e is an element,

i is a gridpoint in e, and

$$\alpha = min \left(\frac{q_{mean}}{q_{mean} - q_{min}}, 1 \right). \tag{2.21}$$

Substituting Equation 2.21 into Equation 2.20, the equation becomes

$$q_i^{new} = q_{mean} + \left(\frac{q_{mean}}{q_{mean} - q_{min}} \right) \left(q_i^{(e)} - q_{mean} \right). \tag{2.22}$$

For the case where $\frac{q_{mean}}{q_{mean} - qmin} > 1$, then $q_{min} > 0$ and $\alpha = 1$. Therefore,

$$q_i^{new} = q_{mean} + (1) \left(q_i^{(e)} - q_{mean} \right), \tag{2.23}$$

which reduces to

$$q_i^{new} = q_i^{(e)}. \tag{2.24}$$

In the case where $\frac{q_{mean}}{q_{mean} - q_{min}} < 1$, then $q_{min} < 0$. Then

$$q_i^{new} = q_{mean} + \left(\frac{q_{mean}}{q_{mean} - q_{min}} \right) \left(q_i^{(e)} - q_{mean} \right). \tag{2.25}$$

If $q_i^{(e)} = q_{min} < 0$, then $q_i^{new} = 0$ [14].

2.7 Time Integrators with Implicit and Explicit Methods

Time integrators are used to advance the solution forward in time. As previously discussed, we use both explicit and implicit time integration.

2.7.1 Explicit Time Integration

Explicit time integration uses information at the current time to compute the solution at the next forward time level. Using the method of lines, our PDE is now an ODE and we must solve the semi-discrete equation

$$\frac{d\mathbf{q}}{dt} = S(\mathbf{q}), \text{where} \tag{2.26}$$

$S(\mathbf{q})$ is the source terms and is in linear form and

$\frac{dq}{dt} = \lambda q$, where $\lambda \neq \lambda(q)$.

To simplify the exposition let us describe how to solve Equation (2.20) explicitly using the Forward Euler (FE) method. FE is written as

$$\frac{\mathbf{q}^{n+1} - \mathbf{q}^n}{\Delta t} = \lambda \mathbf{q}^n. \tag{2.27}$$

We know the information at $\mathbf{q}^n$ in order to solve for the next forward time level $\mathbf{q}^{n+1}$. Solving for $\mathbf{q}^{n+1}$, we get

$$\mathbf{q}^{n+1} = \mathbf{q}^n + \Delta t \lambda \mathbf{q}^n, \tag{2.28}$$

which simplifies further into

$$\mathbf{q}^{n+1} = (1 + \Delta t \lambda)\mathbf{q}^n. \tag{2.29}$$

Explicit methods are easy to implement but, as mentioned in Chapter 1, explicit time integrators are restricted by their time step. To determine the time step for explicit time integrators, we use the Courant-Friedrichs-Lewy (CFL) condition to find the maximum Courant number allowable for the time integrator used.

As mentioned earlier [28], the Courant number is defined as

> A measure of how much information (u) traverses a computational grid cell (Δx) in a given time-step (Δt). For explicit Eulerian methods, the Courant number cannot be greater than 1 because a Courant number greater than 1 means that information is propagating through more than one grid cell at each time step. The time-integrator does not have time to properly interpret what is physically happening and the solution will become unstable. [28]

The Courant number is defined as

$$Courant = max\left(\frac{C\Delta t}{\Delta s}\right)^{e}_{HO} \forall e \in [1, ..., N_e], \tag{2.30}$$

where C is the characteristic speed defined as $C =\mid U + \sqrt{a} \mid$,

U is the velocity in the direction **n** defined as $U = \mathbf{u} \cdot \mathbf{n}$,

a is the maximum celerity of gravity waves in shallow water as defined in Eq (2.10), and

Δs is the grid spacing defined as $\Delta s = \sqrt{\Delta x^2 + \Delta y^2}$ [19].

2.7.2 Implicit Time Integration

Implicit time integration methods find a solution to a PDE using both the current and later state of the system. Unlike explicit time integration, there is no Courant number restriction and therefore implicit methods are not limited by the size of the time step.

One type of implicit method is the Backward Euler's (BE) Method. BE is written as

$$\frac{\mathbf{q}^{n+1} - \mathbf{q}^{n}}{\Delta t} = \lambda \mathbf{q}^{n+1}. \tag{2.31}$$

Rearranging BE in $\mathbf{Ax} = \mathbf{b}$ form we get

$$(I - \Delta t\lambda)\mathbf{q}^{n+1} = \mathbf{q^n}. \tag{2.32}$$

We stated earlier that λ does not equal $\lambda(q)$. Now let us assume that $\lambda = \lambda(q)$. In an explicit method (i.e., FE), this is not an issue and the equation would be written as

$$\frac{\mathbf{q^{n+1}} - \mathbf{q^n}}{\Delta t} = (\lambda\mathbf{q})^n \tag{2.33}$$

and

$$I\mathbf{q^{n+1}} = \mathbf{q^n} + \Delta t(\lambda\mathbf{q})^n. \tag{2.34}$$

While we still are constrained with the Courant number, we know the right hand side of the equation and can solve for $\mathbf{q^{n+1}}$.

When we say $\lambda = \lambda(q)$, for the implicit method, BE becomes

$$\frac{\mathbf{q}^{n+1} - \mathbf{q}^n}{\Delta t} = (\lambda\mathbf{q})^{n+1}. \tag{2.35}$$

In order to solve the matrix $\mathbf{Ax} = \mathbf{b}$, we get BE to be

$$I\mathbf{q}^{n+1} - \Delta t\,(\lambda\mathbf{q})^{n+1} = \mathbf{q}^n. \tag{2.36}$$

We cannot write the solution in the form $\mathbf{A} = \mathbf{I} - \Delta\mathbf{t}\lambda$ because $\lambda = \lambda(q)$, which is a nonlinear equation [28]. Therefore, we use Newton's method for nonlinear systems in order to linearize the problem [13]. Newton's method for nonlinear systems says

$$\mathbf{F}(\mathbf{q}) = \frac{\mathbf{q}^{n+1} - \mathbf{q}^n}{\Delta t} - \mathbf{S}(\mathbf{q}^{n+1}) = 0. \tag{2.37}$$

When $\mathbf{q}$ is the solution, then $\mathbf{F}(\mathbf{q}) = 0$. In order to do this, you must assume you know an initial guess for $\mathbf{q}^{(k)}$, where $k = 0$.

Using a first-order Taylor series expansion (truncation error is $(\Delta q)^2$), we create an outer loop in our algorithm to solve for $\mathbf{F}(\mathbf{q}^{(k+1)})$, which is

$$\mathbf{F}(\mathbf{q}^{(k+1)}) = \mathbf{F}(\mathbf{q}^{(k)}) + \frac{\partial F}{\partial q}(\mathbf{q}^{(k)})(\mathbf{q}^{(k+1)} - \mathbf{q}^{(k)}) = 0 \tag{2.38}$$

where $\mathbf{q}^{(k+1)}$ is the updated solution.

Rearranging $\mathbf{F}(\mathbf{q}^{(k+1)})$, we get

$$\frac{\partial F}{\partial q}(\mathbf{q}^{(k)})\Delta\mathbf{q} = -\mathbf{F}(\mathbf{q}^{(k)}), \tag{2.39}$$

which is now in the linear form $\mathbf{Ax} = \mathbf{b}$ (where $\mathbf{x} = \mathbf{q}^{(k+1)} - \mathbf{q}^{(k)}$), and we can solve the equation.

In order to solve $\frac{\partial F}{\partial q}$, traditionally we would construct the Jacobian. A relatively new method is the Jacobian-free Newton-Krylov (JFNK) method. According to Knoll and Keyes [29], JFNK is defined as

> a nested iteration method consisting of at least two, and usually four levels. The primary levels, which give the method its name, are the loop over the Newton corrections and the loop building up the Krylov subspace out of which each Newton correction is drawn. Interior to the Krylov loop, a preconditioner is usually required, which can itself be direct or iterative. Outside of the Newton loop, a globalization method is often required. This can be implicit time stepping, with time steps chosen to preserve a physically accurate transient. [29]

The JFNK method is extremely complicated and it is important to have a successful Krylov method [28]. The JFNK method is desirable because it does not require building the Jacobian and saves in computational cost. JFNK uses the Newton Method with the addition of GMRES (the Krylov subspace solver) to solve the linear problem $\mathbf{Ax} = \mathbf{b}$.

As shown by [8], our Krylov subspace is defined as

$$\kappa_n =< \mathbf{b}, \mathbf{Ab}, ..., \mathbf{A}^{n-1}\mathbf{b} >=< \mathbf{q_1}, \mathbf{q_2}, ..., \mathbf{q_n} >\subseteq \mathbb{C}^m \tag{2.40}$$

and can also be written in matrix form, where

$$\kappa_n = \begin{vmatrix} \mathbf{b} & \mathbf{Ab} & \dots & \mathbf{A}^{n-1}\mathbf{b} \end{vmatrix}. \tag{2.41}$$

In order to solve $\frac{\partial F}{\partial q} = J$, we write

$$\frac{\partial F}{\partial q} = \frac{F(\mathbf{q}^{(k+1)} - F(\mathbf{q}^{(k)})}{\mathbf{q}^{(k+1)} - \mathbf{q}^{(k)}}. \tag{2.42}$$

We determine a stopping criteria based on an acceptable tolerance, ϵ, to solve for $\Delta\mathbf{q}$ such that

$$\frac{\partial F}{\partial q} = \frac{F(\mathbf{q} + \epsilon\Delta\mathbf{q}) - F(\mathbf{q})}{\epsilon\Delta\mathbf{q}}. \tag{2.43}$$

As demonstrated in [28], rearranging the equation we get

$$\frac{\partial F}{\partial q}\Delta\mathbf{q} = \frac{(F(\mathbf{q} + \epsilon\Delta\mathbf{q}) - F(\mathbf{q}))\,\Delta\mathbf{q}}{\epsilon\Delta\mathbf{q}}, \tag{2.44}$$

which reduces to

$$\frac{\partial F}{\partial q}\Delta\mathbf{q} = \frac{F(\mathbf{q} + \epsilon\Delta\mathbf{q}) - F(\mathbf{q})}{\epsilon}. \tag{2.45}$$

In conclusion, even though we can have infinitely large time steps using implicit methods we are still constrained by accuracy. While the system may be stable and there is no CFL condition, we cannot just choose an infinitely large time step and expect to have an accurate solution [28]. There will continue to be a growing demand for solving PDEs through computational models. In our problem, solving the SWE will require continuing improvements to the order of accuracy and processing time.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 3:
Verification and Test Cases

The overall objective when solving numerical methods computationally is to obtain a solution. Not only is it important to have a solution, but also the solution must be unique and accurate. Along with accuracy, another criterion one strives for in computation is stability. In other words, making small changes in the initial data or conditions should not dramatically change the solution. In addition, we look for the rate of convergence for the solution. To solve a PDE, we use a discretization method to represent or approximate a continuous problem. It is important that convergence is obtained. Lastly, your results should provide a valid solution that makes practical sense to a realistic problem you are solving [2]. When you have stability, convergence, and a result that makes practical sense, then you can accept that there is a unique and accurate solution, which is also known as a well posed problem [30].

While a unique and accurate solution is important, it is also important to have a computationally efficient model. Models that have large domains can turn into time intensive computations. One way to reduce computation time is by changing the grid resolution. Depending on the type of problem you are solving you do not need precise resolution at every point in your domain. This is often the case when solving the SWE where you have a large domain but need resolution only every few meters to solve the problem. Let us define grid resolution as follows:

$$Resolution = \frac{X_{max} - X_{min}}{N_{points}}, \tag{3.1}$$

$$N_{points} = N_{elx} * N_{op}, \tag{3.2}$$

where N_{elx} is the number of elements in the x direction, and

N_{op} is the degrees of freedom of the higher order polynomial [14].

Prior to running a model, it is critical to verify the method you are using to solve that model. Verification of models is done through test cases, and these cases are compared to test cases that have been previously performed. For our model, we used three verification cases of the DG method with a nodal basis on quadrilaterals. Since no experimental solution was feasible, the models were compared to other numerical simulations. Specifically, these cases were modeled and compared to three Balzano case studies, which simulate wetting and drying in one-dimensional shallow water flow [26].

The Balzano test cases were first used to compare wetting and drying methods for finite difference methods. Balzano used a one-dimensional domain and was evaluating overall accuracy [31]. His models compared large time steps with implicit schemes on an Arakawa-C grid using a semi-implicit, Eulerian-Lagrangian model with the goal of having no loss of local mass due to wetting and drying. Balzano found that his results did yield accurate solutions but had long computational times associated with them, which is most likely due to the wetting and drying method he used [26].

The three test cases we used represented three different forms of bathymetry and represented a tide that swells and ebbs. By changing the bathymetry, the models increased in complexity. All three test cases were completed in a basin width of 13,800 meters and a depth of 5 meters. There was an open boundary at one end and no-flow boundary on the other end. We ran the model over a period of 43,200 seconds (12 hours) with a time step 10 seconds and a time restart of 900 seconds (0.25 hours). The water began at a height of two meters above the bathymetry. Over the first 21,600 seconds (6 hours), the water level was lowered at a consistent rate to a total of four meters. After the water level was lowered, then the water rose to the original water surface level over the last 21,600 seconds (6 hours). Twenty-five elements were used in the x-direction, and two elements were used in the y-direction with a fourth order polynomial.[2] This provided a grid resolution of 138 meters.

For time integrators, both an implicit and explicit method were used for each test case. It was important to run the two time integrators to ensure we had the same results. Having the same results allowed us to verify the stability and accuracy of our method. We evaluated the difference in values for the implicit and explicit method for both water surface level and water velocity. To compute the error, the following equation was used:

---

[2] Our test cases were in 1-D. Since our code is written in 2-D, we used more than one element in the y-direction.

$$\text{Error Norm} = \frac{\text{Norm(Explicit Method} - \text{Implicit Method)}}{\text{Norm(Explicit Method)}} \tag{3.3}$$

According to [8], a norm is a function (or a mapping)

$$|| \cdot ||\colon \mathbb{C} \rightarrow \mathbb{R}, \tag{3.4}$$

which provides a real-valued length to each vector [8].

The three most common norms (1-Norm, 2-Norm, and ∞-Norm) were used for finding the error norm. These norms are defined as

$$|| \ x \ ||_1 = \sum_{i=1}^{m} | \ x_i \ |, \tag{3.5}$$

$$|| \ x \ ||_2 = \left(\sum_{i=i}^{m} | \ x_i \ |^2 \right)^{\frac{1}{2}} = \sqrt{x * x}, \tag{3.6}$$

$$|| \ x \ ||_\infty = max_{1 \leq j \leq m} \ | \ x_i \ | \ . \tag{3.7}$$

where x is a m-vector [8].

3.1 Case 1: Linear Bathymetry

The first case used a linear bathymetry (Figure 3.1). The slope had a uniform bottom, and over the 12-hour simulation there were two phases in the flow [31]. First, in the ebb phase, the tide was draining away from the shore and then after 6 hours, began to swell and return to its original position. Figure 3.2 represents Case 1 using a fully implicit time integrator. As you can observe, there was no significant difference with the two time integrators, and both methods provided stable solutions. Table 3.1 and Table 3.2 show the error norms for the surface height and velocity over the 12-hour time period. The error was approximately 1×10^{-1} (less than 1 centimeter difference for a domain of 13,800 meters).

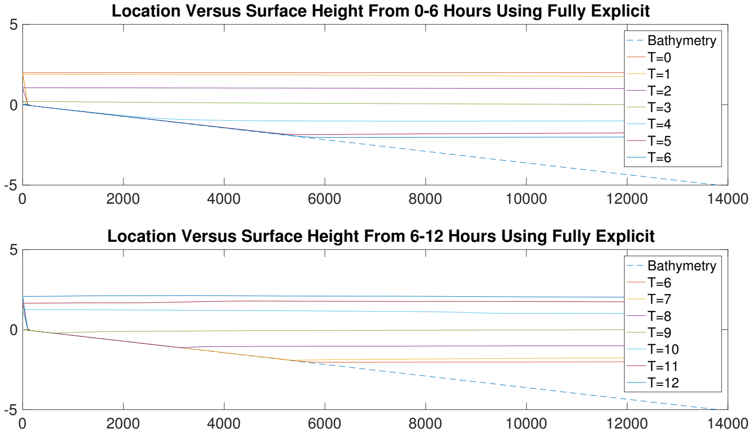


Figure 3.1: Case 1: Testing simulation in 1D with linear bathymetry using fully explicit time integration.

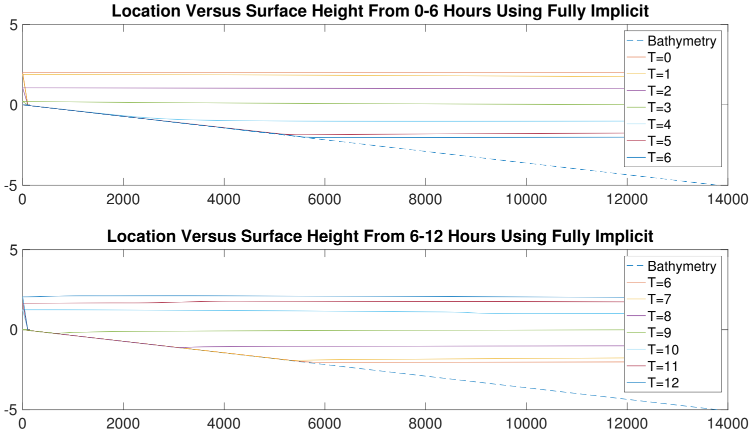


Figure 3.2: Case 1: Testing simulation in 1D with linear bathymetry using fully implicit time integration.

Table 3.1: Surface height error norm for linear bathymetry

	T=0	T=2	T=4	T=6	T=8	T=10	T=12
1-Norm	0	1.00E-03	1.11E-03	1.25E-04	2.00E-03	1.19E-03	1.2E-03
2-Norm	0	1.00E-03	1.12E-03	2.87E-04	2.6E-03	2.27E-03	2.1E-03
Infinity-Norm	0	1.00E-03	1.13E-03	1.6E-03	1.03E-02	9.6E-03	1.016E-03

Table 3.2: Velocity error norm for linear bathymetry

	T=0	T=2	T=4	T=6	T=8	T=10	T=12
1-Norm	0	1.80E-04	7.16E-04	1.13E-02	1.27E-02	6.2E-02	4.8E-03
2-Norm	0	1.84E-04	1.3E-03	2.99E-02	3.36E-02	8.1E-03	9.6E-03
Infinity-Norm	0	2.18E-04	5.5E-03	1.008E-01	1.747E-01	2.36E-02	5.22E-02

3.2 Case 2: Linear Bathymetry with Shelf

The second case we modeled was slightly more complex (Figure 3.3). The concept of a tide ebbing and swelling remained the same and the only change was in the basin's bathymetry. Unlike Case 1, the slope did not remain linear. Instead, we used a changing slope to simulate a "shelf" on the basin floor. We used a uniformly sloped bathymetry from 0 to 3,600 meters. The slope then leveled to represent a shelf, which ran from 3,600 meters to 4,800 meters. There was again a pitched slope from 4,800 meters to 13,800 meters.

You can see as the tide receded around hour four that the tide was impacted as it approached the shelf along the bathymetry. The water level was no longer even. As reflected by a small wave as seen in Figure 3.3, this change reflects the physical behavior we would expect for tidal flow along a bathymetry with a shelf.

When the simulation was run with a fully implicit time integrator, there were no dramatic changes to the solution. Figure 3.4 represents the results for Case 2 using fully implicit integration. The results of the error norm were very similar to the linear bathymetry model and remained within 1×10^{-1} magnitude (less than 1 centimeter difference). Table 3.3 and 3.4 shows the error norm for surface height and velocity over the 12 hour period.

Table 3.3: Surface height error norm for linear bathymetry with a shelf

	T=0	**T=2**	**T=4**	**T=6**	**T=8**	**T=10**	**T=12**
1-Norm	0	1.00E-03	2.10E-03	1.30E-03	3.60E-03	2.70E-03	1.50E-03
2-Norm	0	1.00E-03	3.80E-03	2.30E-03	5.50E-03	4.70E-03	2.40E-03
Infinity-Norm	0	1.00E-03	1.80E-03	6.90E-03	1.82E-03	2.00E-02	9.80E-03

Table 3.4: Velocity error norm for linear bathymetry with a shelf

	T=0	**T=2**	**T=4**	**T=6**	**T=8**	**T=10**	**T=12**
1-Norm	0	1.94E-04	7.10E-03	3.48E-02	3.76E-02	1.02E-02	1.37E-02
2-Norm	0	2.01E-04	2.06E-02	5.93E-02	6.73E-02	1.49E-02	1.67E-02
Infinity-Norm	0	4.38E-04	5.706E-02	1.01E-02	2.09E-02	4.89E-02	3.90E-02

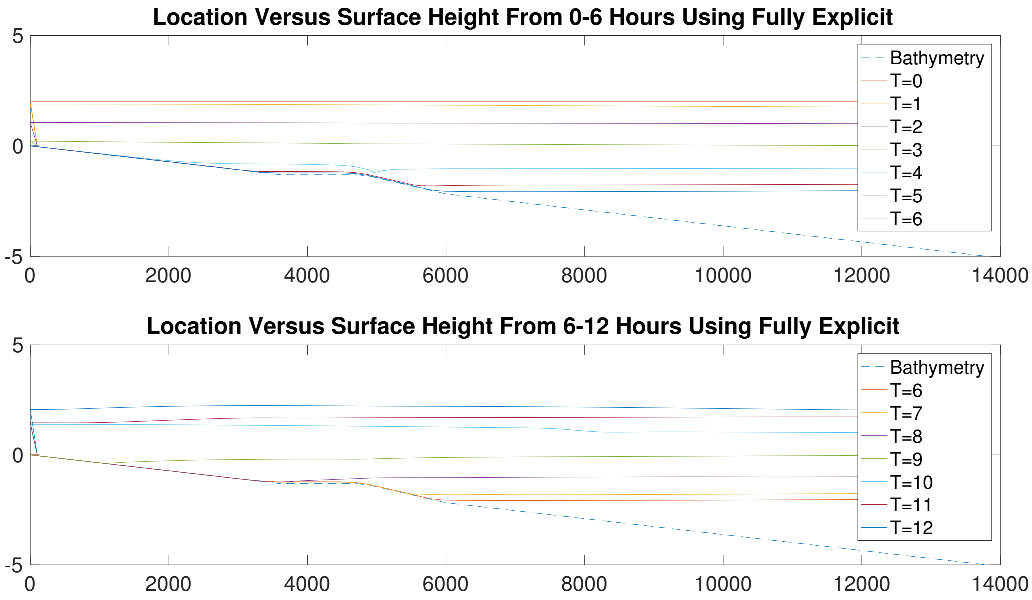


Figure 3.3: Case 2: Testing simulation in 1D with changing sloping bathymetry to represent a shelf using fully explicit time integration.

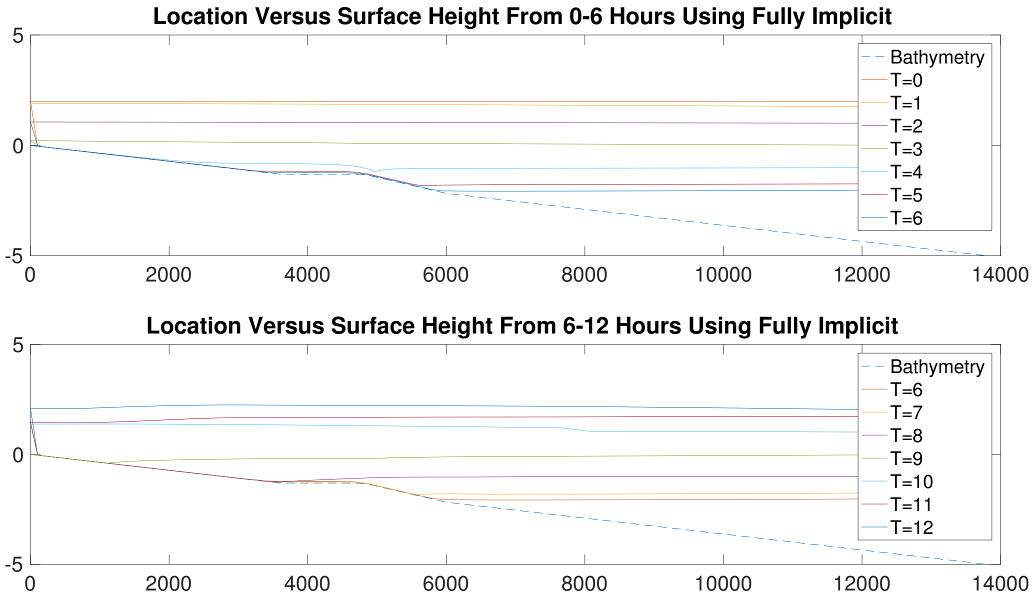


Figure 3.4: Case 2: Testing simulation in 1D with changing sloping bathymetry to represent a shelf using fully implicit time integration.

3.3 Case 3: Linear Bathymetry with Tidal Pool

The third and final case was the most complex because we needed multiple changing slopes in bathymetry to simulate a tidal pool in a basin (Figure 3.5). The external force of the tide ebbing and flowing remained consistent with Case 1 and Case 2. The new bathymetry was a linear slope from 0 to 3,600 meters. Then from 3,600 meters to 48,000 meters, the slope became positive to represent a tidal pool. From 4,800 meters to 13,800 meters the slope returned to a negative linear slope.

In Figure 3.5, you can see the tide reacting with the tidal pool in the basin, similar to Case 2 when the tide reacted with the shelf. The tidal reaction with the bathymetry is even more prominent as the tide shifted from receding to swelling and returned to initial surface levels. This interaction between the water in the bathymetry illustrates what we would expect to happen.

This case was the most computationally intensive of the three tests due to being a more complex problem due to having positive and negative slopes. The simulation was able to run for fully implicit (Figure 3.6) and the results were similar to the fully explicit method.

Table 3.5 and Table 3.6 shows the error norm for the surface height and velocity.

By running the three test cases, we were able to confirm that our method provides stable results that converge to accurate and unique solutions. This verification will allow us to run other simulations. In the next chapter, a new simulation will be done with a different bathymetry and acting force. Instead of a tide ebbing and swelling, we will simulate an acting force of a wave, which will represent a storm surge.

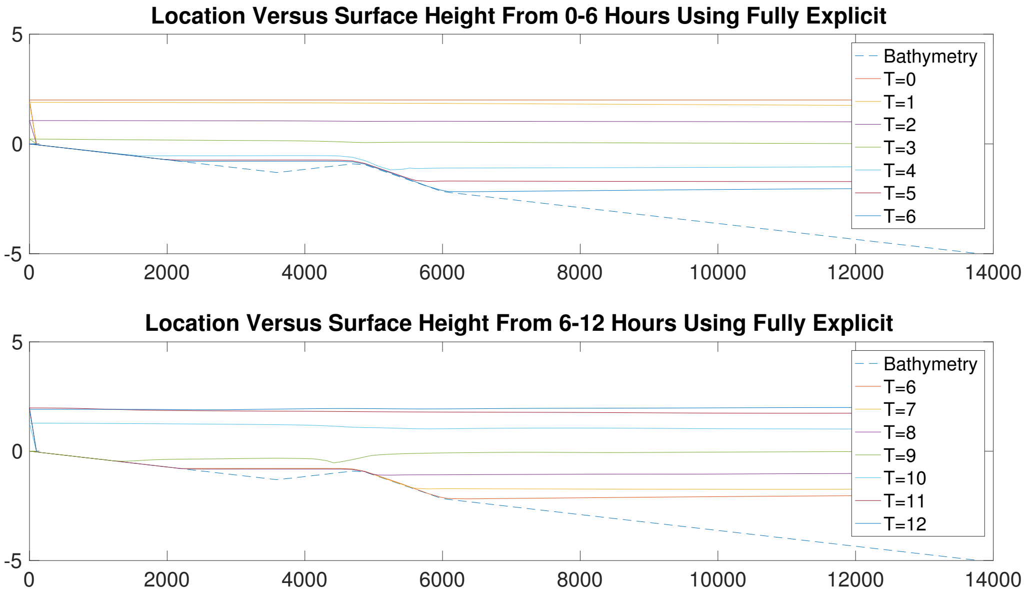


Figure 3.5: Case 3: Testing simulation in 1D with changing sloping bathymetry to represent a tidal pool using fully explicit time integration.

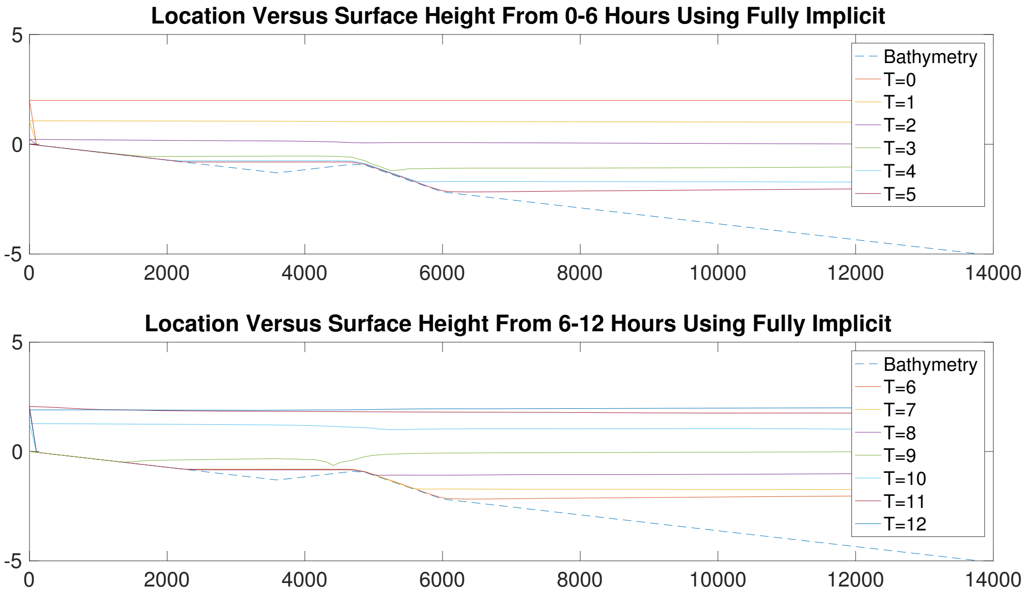


Figure 3.6: Case 3: Testing simulation in 1D with changing sloping bathymetry to represent a tidal pool using fully implicit time integration.

Table 3.5: Surface height error norm for linear bathymetry with a tidal pool

	T=0	T=2	T=4	T=6	T=8	T=10	T=12
1-Norm	0	1.00E-03	5.50E-03	3.20E-03	7.10E-03	9.00E-03	5.00E-03
2-Norm	0	1.00E-03	7.20E-03	4.90E-03	1.12E-02	1.24E-02	8.40E-03
Infinity-Norm	0	1.10E-03	1.96E-02	8.80E-03	2.44E-02	4.90E-02	2.48E-02

Table 3.6: Velocity error norm for linear bathymetry with a tidal pool

	T=0	T=2	T=4	T=6	T=8	T=10	T=12
1-Norm	0	2.23E-04	2.39E-01	1.16E-01	6.88E-02	1.96E-02	3.64E-01
2-Norm	0	2.81E-04	3.95E-02	1.80E-02	1.41E-02	2.79E-02	5.58E-02
Infinity-Norm	0	1.20E-03	8.08E-02	3.58E-01	3.58E-01	4.88E-01	1.62E-01

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 4:
Simulation of A Storm Surge in La Push, Washington

After verifying the correctness of our model with three test cases, we can use a real-world case study to analyze the changes in momentum flux from a wave due to altering the bathymetry. Our real-world case study will compare the momentum flux for two different types of bathymetry based on field data from La Push, Washington. We also analyze two different tide levels for each bathymetry.

4.1 Project Description

We simulate the effects of a storm surge on a USACE beach restoration project within the Quileute Nation's reservation land, which is located on the northwest coast of the Olympic Peninsula [15]. Refer to Figure 4.1 for a detailed location of the project site. A beach restoration project consists of placing materials such as sand, sediment and/or large armor stones along a shoreline to reduce damage from different types of storms to include storm surges. The intent of a beach restoration project is to prevent overtopping waves from breaching the mainland and causing damage and erosion.

The restoration project in La Push is important because, when there is a storm surge that breaches the land, the local navigation channel becomes filled with sediment due to erosion. The local navigation channel is important for both the United States Coast Guard (USCG) and the Quileute Nation. The USCG has a local station there and uses the channel for search and rescue operations along the Olympic Peninsula. Their station is the only one between Westport, Washington and Neah Bay, Washington. The Quileute Nation relies on the channel as a harbor for subsistence fishing [15].

Along the Olympic Peninsula, winter and spring storms cause significant erosion and damage in La Push. The damage caused by these storms motivated the USACE to provide beach restoration under the authority Public Law (PL) 84-99 [15]. PL 84-99 grants authority to USACE from Congress to react to emergencies due to flooding and provides them with the capabilities to repair and rehabilitate flood control projects [32].

The simulations demonstrates a wave breaking on a shoreline with existing bathymetry and a wave breaking on a shoreline with a new bathymetry based on the plans for the restoration project. The main inputs for the model are the existing and new bathymetry, the amplitudes of waves from existing field data, boundary conditions, field data measurements of low and high tides, the number of elements for resolution, and time step.

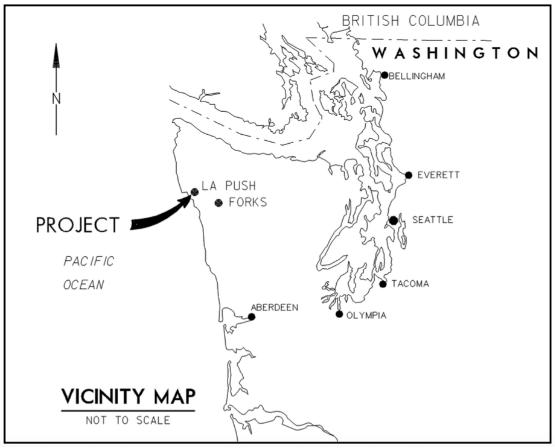
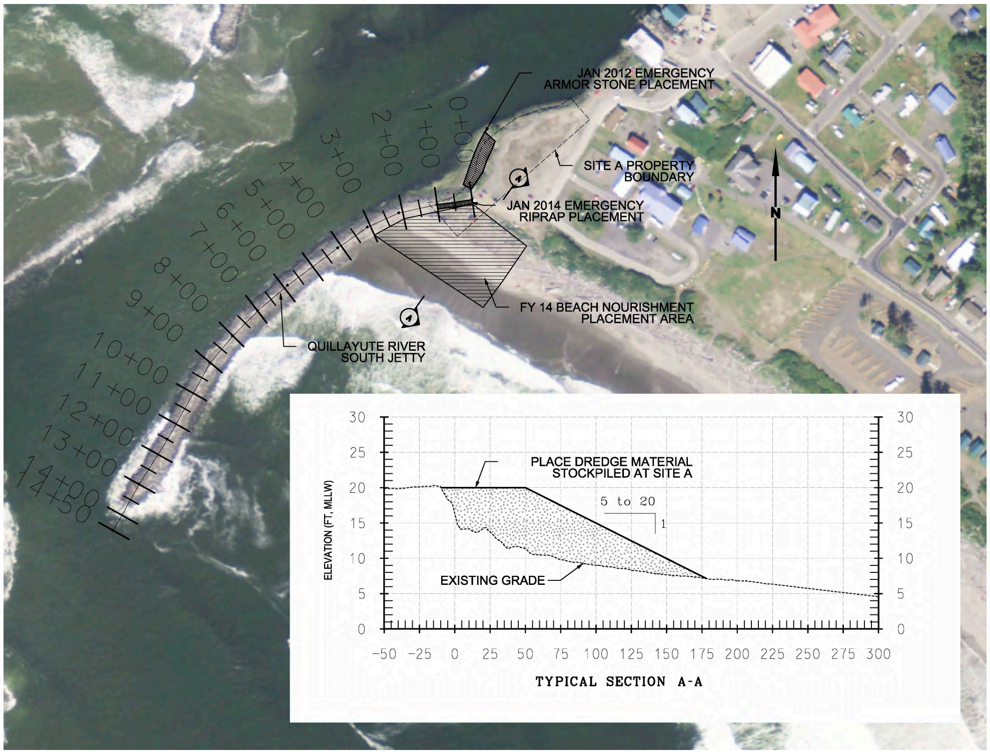

BRITISH COLUMBIA
WASHINGTON
N
BELLINGHAM
EVERETT
LA PUSH
FORKS
SEATTLE
PROJECT
PACIFIC
OCEAN
TACOMA
ABERDEEN
OLYMPIA
VICINITY MAP
NOT TO SCALE

Figure 4.1: Location of beach restoration in La Push, Washington. Source: [33].

4.1.1 Bathymetry

The USACE gathered real bathymetry data on the shoreline in La Push, Washington. Figure 4.2 represents a typical cross-sectional area of the existing bathymetry and the planned bathymetry for the beach restoration used in our model.



JAN 2012 EMERGENCY ARMOR STONE PLACEMENT
SITE A PROPERTY BOUNDARY
JAN 2014 EMERGENCY RIPRAP PLACEMENT
FY 14 BEACH NOURISHMENT PLACEMENT AREA
QUILLAYUTE RIVER SOUTH JETTY
N
0+00
1+00
2+00
3+00
4+00
5+00
6+00
7+00
8+00
9+00
10+00
11+00
12+00
13+00
14+00
14+50
PLACE DREDGE MATERIAL STOCKPILED AT SITE A
5 to 20
1
EXISTING GRADE
ELEVATION (FT, MLLW)
0
5
10
15
20
25
30
-50
-25
0
25
50
75
100
125
150
175
200
225
250
275
300
TYPICAL SECTION A-A

Figure 4.2: Cross-sectional of existing and future bathymetry of La Push, Washington. Source: [15].

The USACE cross-sectional data was then replicated in our model and used as the foundation for modeling our real-world test cases. To simulate the reproduction of the model we built two different bathymetries in our code. Our bathymetries were built as part of our initial conditions and in the construction of our grid geometry in our model. We established an x-max in our domain to replicate the distance the wave would travel. The real world bathymetry was then broken into multiple sections where we fitted line segments and parabolic functions based on the magnitude of changing slopes to the real world bathymetry for replication. While we were not able to exactly replicate the field data we attempted to make a closest fit to each section of the bathymetry. Figure 4.3 represents the reproduction of the old and new bathymetry.

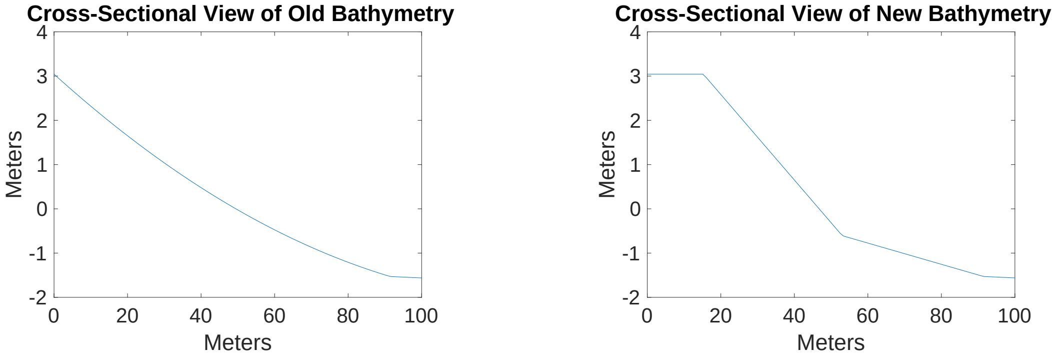


Figure 4.3: Reproduction of old and new bathymetry

4.1.2 Boundary Conditions and Manning's Coefficient

To simulate a storm surge we treated the shoreline as a fixed wall boundary. We used a no-flux boundary as Alevras did in his masters thesis on a real-world tsunami in the Indian Ocean [7]:

$$\mathbf{n} \cdot \mathbf{u} = 0. \tag{4.1}$$

We used a finite domain with fixed boundaries at the far left and right and evaluated the initial wave that hit the left bounded wall of the model. Since it was a computationally expensive calculation, we needed to approximate when the initial interaction of the wave with the left boundary wall would occur prior to running the simulation. To do this, we used the wave speed formula for estimating wave time travel. The equation used for the wave speed formula is

$$speed = \sqrt{g * H} \tag{4.2}$$

where H is the water depth and

g is acceleration due to gravity [34].

After we find the speed of the wave with respect to gravity and height we can find the approximate time a wave will take to reach the shoreline with the formula

$$time = \frac{L}{speed}, \tag{4.3}$$

where L is the distance the wave will travel.

For our model, we used data collected from a buoy with a water depth of 114.3 meters and 30,000 meters from the shoreline [35]. Using these parameters, we would expect the wave to reach the shoreline in 15 minutes at the earliest.

Based on [36], we used a Manning's roughness coefficient of 0.02 to replicate the friction from the sand interacting with the wave on the ocean floor. Another assumption we made was that the sand material was uniform across the bed of the ocean floor and therefore, that Manning's coefficient never changed.

4.1.3 Wave Amplitude and Base

We determined the wave amplitude based on a year's worth of historical data on waves from the National Data Buoy Center, which is part of the National Oceanic and Atmospheric Administration (NOAA). We analyzed data from Buoy Station 46041 (near Cape Elizabeth, Washington), which is located at 47.353 N 124.731 W [35]. See Figure 4.4 for the exact location. The station is approximately 30 kilometers from the project site. This buoy data from 2014 was substantiated by an independent field study contracted by USACE. Researchers involved in this independent field study took measurements of waves, water levels, and currents at a location offshore to the mouth of the Quillayute River near La Push, Washington (47.536N, 124.392W). Figure 4.5 shows the location of their study [15].

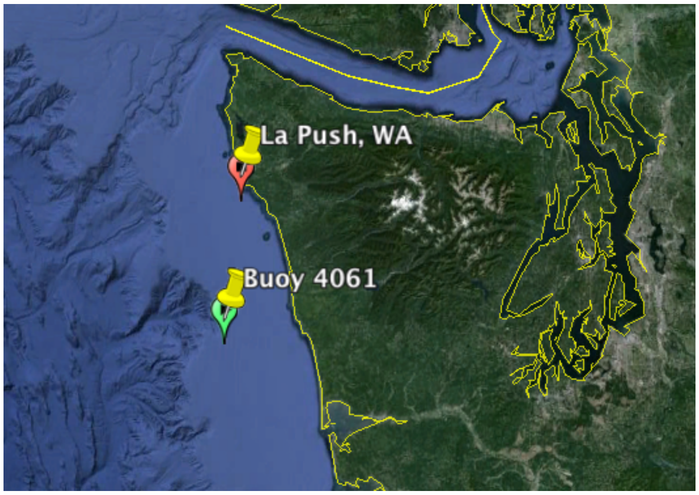

La Push, WA
Buoy 4061

Figure 4.4: Location of Station 46041 with respect to La Push, Washington. Adapted from data in [37].

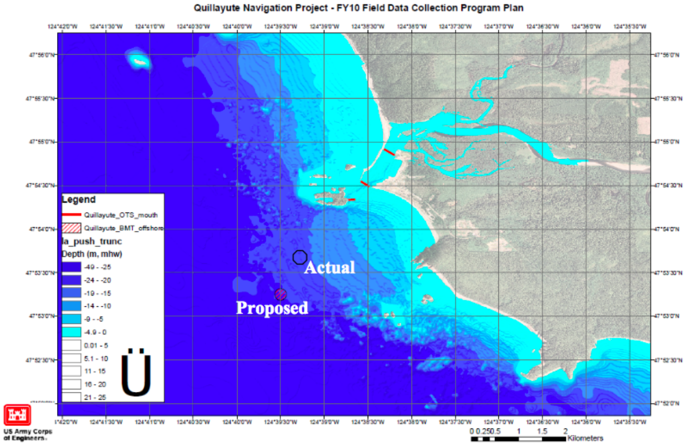

Quillayute Navigation Project - FY10 Field Data Collection Program Plan
Legend
Quillayute_OTS_mouth
Quillayute_BMT_offshore
la_push_trunc
Depth (m, mhw)
-49 - -25
-24 - -20
-19 - -15
-14 - -10
-9 - -5
-4.9 - 0
0.01 - 5
5.1 - 10
11 - 15
16 - 20
21 - 25
Actual
Proposed
US Army Corps of Engineers
0 0.250.5 1 1.5 2
Kilometers

Figure 4.5: Location of proposed and actual location of field measurements conducted by USACE for La Push, Washington. Source: [15].

Figure 4.6 and Figure 4.7 display a year's worth of data from Station 46041, specifically the maximum wave height and the hourly wave period [35].

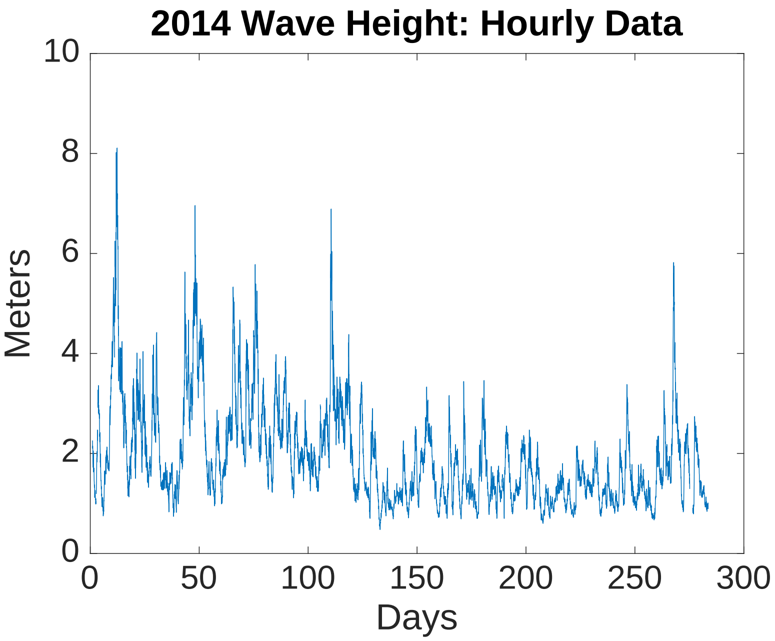


Figure 4.6: Maximum wave heights At Station 46041 in 2014. Adapted from data in [35].

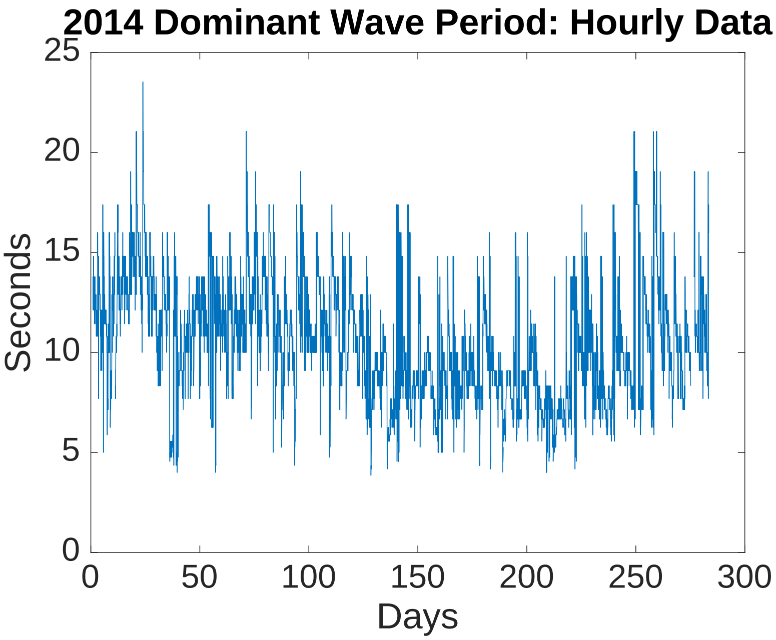


Figure 4.7: Period between maximum waves At Station 46041 in 2014. Adapted from data in [35].

According to the data from Station 46041, the maximum wave height was close to 9 meters with a period between 5 to 20 seconds between waves. The data was very similar to the USACE field data, which showed the maximum wave height as approximately 11 meters and an average maximum wave height of approximately 6 meters [15]. For our simulation, we modeled 11-meter waves to account for the worst-case scenario. Station 46041 is 30 kilometers from the project site; therefore, we needed to extend our domain to accurately reflect the starting point of the wave. The water depth at Station 46041 is 114.3 meters. Since we do not have data on the bathymetry for the entire domain we made an assumption that the bathymetry is linear 100 meters from the shoreline to Station 46041 (30,000 meters from the shoreline) [35]. While this is a broad assumption, with limited information this was our best solution. In addition, we are primarily concerned with detailed data near the shoreline where the changes in new and old bathymetry were occurring.

Waves in the ocean have some inherent properties to include a relationship between wavelength, period, and wave speed. We used these relationships to determine the base of the wave. From the *Environmental Systems and Introduction Text*, we used the shallow water wavelength equation, which is appropriate for storm surges since they typically have long wavelengths. The shallow water wavelength equation is defined as

$$\lambda = 1.56T^2 \tag{4.4}$$

where T is the period between waves [38].

Using the wave period data from Station 46041, we determined a total wavelength of 40 meters, making the wave base 20 meters.

4.1.4 High Tide Versus Low Tide

We wanted to evaluate the effects of storm surges on the old and new bathymetry based on different sea levels that occur in the local area. In particular, we wanted to look at the extreme high and low tides in La Push, Washington, which resulted from the monthly lunar cycle. Based on annual field data from U.S. Harbors, La Push has an extreme high tide of 2.96 meters and an extreme low tide of 0.67 meters [39]. To simulate the high and low tides in the model, we shifted the bathymetry to correspond with the changing vertical distance of the water level with respect to the basin floor.

4.1.5 Resolution and Time Step

Given our large domain (30,000 meters), there was a large computational cost required to run the model. While we wanted an accurate and unique solution, we also wanted to have a computationally efficient model. To reduce the computational cost, we modified the grid resolution. To maintain high-order accuracy, we still used a fourth order polynomial and two elements in the y-direction. For our model, we were only concerned with the effects of the wave on the bathymetry for the last 100 meters because this location is where the changes between the old and new bathymetry occur. Also, since we were simulating a wave that has an amplitude of 11 meters and wavelength 40 meters (wave base of 20 meters), we only needed a resolution of 4 meters in the domain. With a resolution of over 4 meters with a domain of 30,000 meters we needed 1880 elements in the x-direction, which means that we required 7520 points to run the model.

After we ran the model, we realized that the number of x-elements did not accurately represent the bathymetry. There were discrepancies in the new bathymetry occurring at the last 100 meters where multiple changes in the bathymetry slope were occurring. With 7520 points, the model had a hard time transitioning from changing bathymetry slopes causing distortions in the model to include extra line segments. The discrepancies caused us to change the resolution from 1880 elements in the x-direction to 5640 elements that resulted in a total of 22560 points. While the increase in elements was necessary, the increase by 15040 elements more than doubled the computational time to run the model. You can see the difference in resolution in Figure 4.8. In Figure 4.8, you see discrepancies in the bathymetry at 15 meters and 53 meters in the domain. At 15 meters, the bathymetry should change from a slope of 0 to 0.0966 and at 53 meters the bathymetry should change slope from 0.0966 to 0.024. Using 7520 points, there are variances in the slope causing extra line segments to display due to lack of points. With 22560 points you do not see these extra line segments. It was important to have a higher resolution otherwise when the wave approaches the bathymetries we would not be able to get an accurate representation of how effective the bathymetries were in response to a storm surge.

Lastly, we decided our model should run with a time step of 0.05 seconds and time restart of every 45 seconds (0.75 minutes) both implicitly and explicitly for a total of 2100 seconds (35 minutes). For implicit integration, we could have used a much larger time step based on our validation of our test cases in Chapter 3, but decided to keep the same time step as we

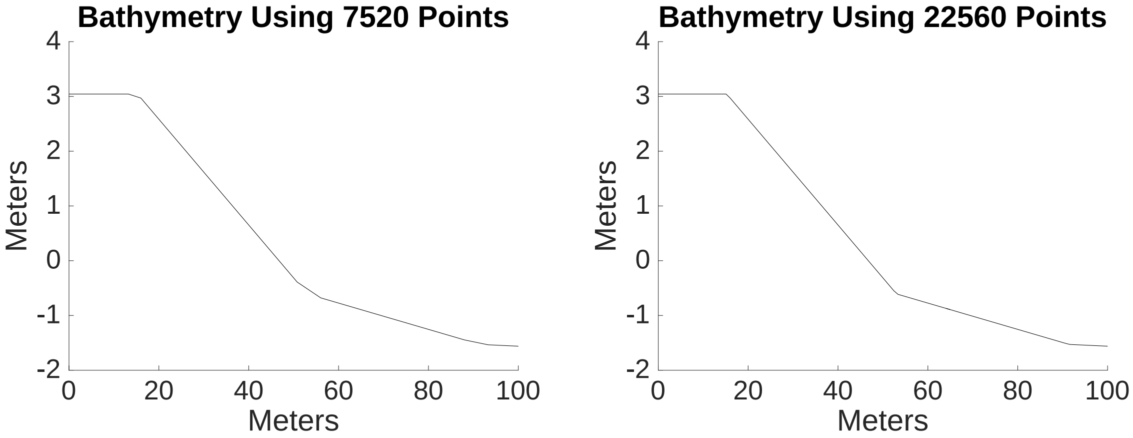


Figure 4.8: Difference in bathymetry resolution using 7520 and 22560 points.

used for explicit integration. We kept the same time step in order to compare the difference in error between implicit and explicit integration. We wanted to see how different the implicit solution would be from the explicit. If we chose a different time-step, then we would introduce errors of the order of the time-integrator accuracy and would not be able to compare implicit with explicit. We choose a time restart of every 45 seconds based on the velocity of the wave. If we used a larger time restart than 45 seconds we would risk not effectively capturing the simulation. For every 45 seconds, the model provided a new frame of data on the wave to include the location along the shoreline, wave height, and velocity of the wave.

4.2 Metric Used to Measure Effectiveness of Bathymetry

To validate the effectiveness of the beach restoration project, we used a similar method as [40] and found the momentum flux of the waves to determine their strength and destructive power. We looked at both the momentum flux of an 11-meter wave at high and low tide for the old and new bathymetry. Conceptually, momentum flux is similar to kinetic energy. While kinetic energy is defined as momentum per mass, for momentum flux we use the total column height of the water as the role of mass. The equation used for momentum flux is

$$\frac{1}{2}\vartheta^2 H, \tag{4.5}$$

where ϑ is velocity of the wave, and

H is the total water column height ($H = h_b + h_s$).

The distance from the bathymetry to the mean level of the surface water is h_b and h_s is the distance from the mean level of the surface water to the surface height [41].[3]

More specifically, we analyzed the momentum flux of the wave just prior to the wave reaching the left boundary. The momentum flux was graphed with respect to time and compared against different bathymetries.

4.3 Results From The Model

We first looked at the total momentum flux for the entire 30,000 meter domain. We took the 2-norm of the momentum flux for each time frame in the simulation to show the total momentum flux. Across the entire domain, there was very little difference between the old and new bathymetry as seen in Figure 4.8. The lack of difference makes practical sense due to the main difference in bathymetry occurring only in the last 100 meters. Therefore, we also looked at the total momentum flux in the last 100 meters.

---

[3] This equation is useful for predicting the amount of energy near the coastline because the wave height (H) will increase near shallow water even while the velocity (ϑ) decreases.

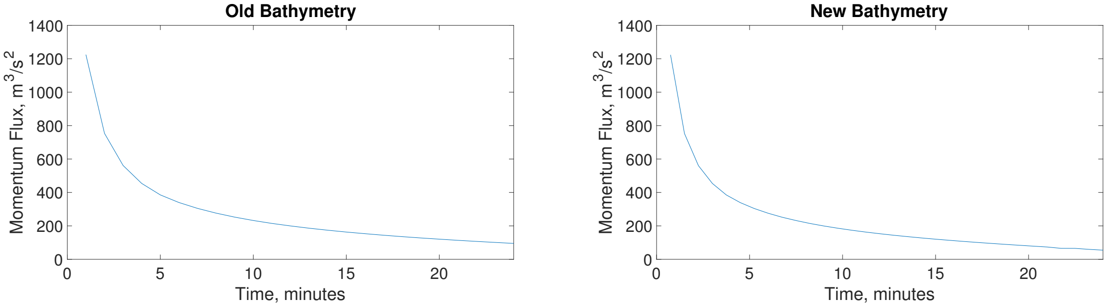


Figure 4.9: Total norm momentum flux for the entire computational domain (30,000 meters).

4.3.1 High Tide

Looking at the surface height of the wave as it approaches the shoreline you can see the wave is approximately 2 meters high for both the old and new bathymetry (See Figure 4.9). We determined when the wave hit the shoreline first by estimating the arrival of the wave using the wave speed formula. We then refined our wave arrival time to shore by running a time lapse video of the output frames from our model. Using our time lapse video, we determined, in high-tide conditions, that the wave hits the shore line at 21 minutes. Analyzing Figure 4.9, you see little difference between the surface height for the old and new bathymetries. One difference between the two is that as the wave approaches the old bathymetry the slope of the wave is significantly steeper than in the new bathymetry. This is indicative of a hydraulic jump or bore occurring. The oncoming wave of the new bathymetry is less vertical, which indicates the wave is not as strong.

It is not until you look at the momentum flux that you notice a more significant impact from the changed bathymetry. As seen in Fig 4.10, the new bathymetry has less momentum flux than the old and therefore the changes are more effective at mitigating the damage from storm surges. The peak momentum flux for the old bathymetry was 3.9 $\frac{m^3}{s^2}$ compared to the new bathymetry with a peak of 2.9 $\frac{m^3}{s^2}$, which is a difference of 1.0 $\frac{m^3}{s^2}$.

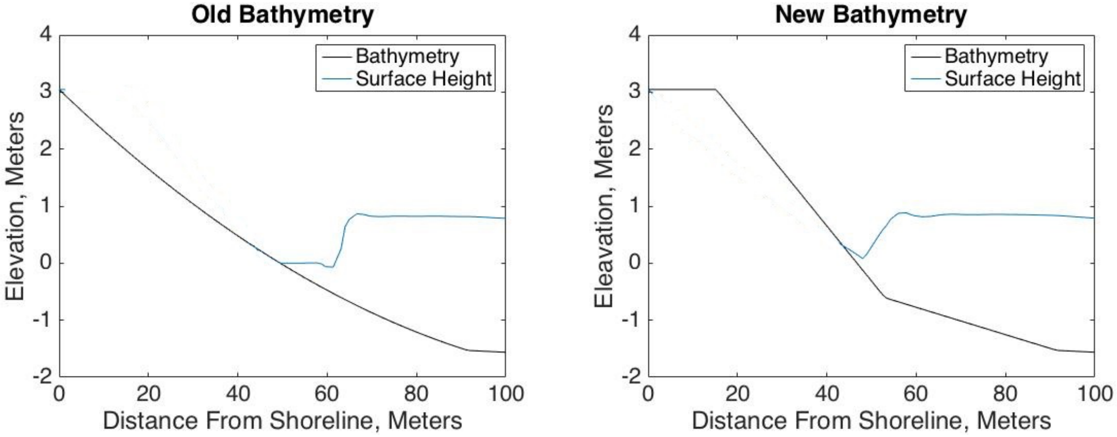


Figure 4.10: Surface height of the wave prior to reaching the shoreline for old and new bathymetries during high-tide.

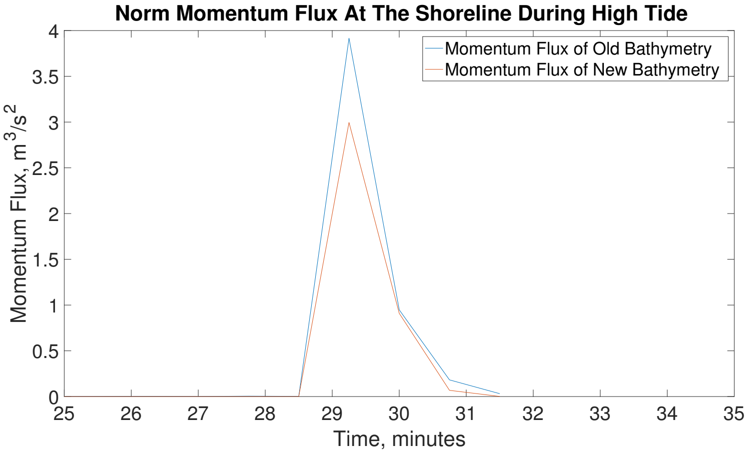


Figure 4.11: Momentum flux of the old and new bathymetries along the closest 100 meters to shoreline during high-tide.

4.3.2 Low Tide

For low tide, the surface height of the wave as it approaches the shore is about 1 meter smaller than for the high tide (See Figure 4.11). Similar to the high tide model, we used a time lapse video to determine when the wave hit the shoreline. The wave took approximately 23 minutes to reach the shore. Because there is less water during low tide, the velocity of the wave was slower and therefore took longer to arrive at the shore than during high tide conditions. Both the old and new bathymetry have similar wave heights but you can see that the base of the wave for the new bathymetry is not as large compared to the old bathymetry. While both bathymetries have similar wave heights, there is a significant difference in momentum flux from the different bathymetries (See Figure 4.12). The old bathymetry has a maximum momentum flux of 1.12 $\frac{m^3}{s^2}$ compared to the new bathymetry with a maximum momentum flux of 0.41 $\frac{m^3}{s^2}$, which is a difference of 0.79 $\frac{m^3}{s^2}$. The scale of variance of momentum flux from the low tide compared to the high tide means that the new bathymetry is more effective during a storm surge in low-tide conditions compared to high-tide conditions.

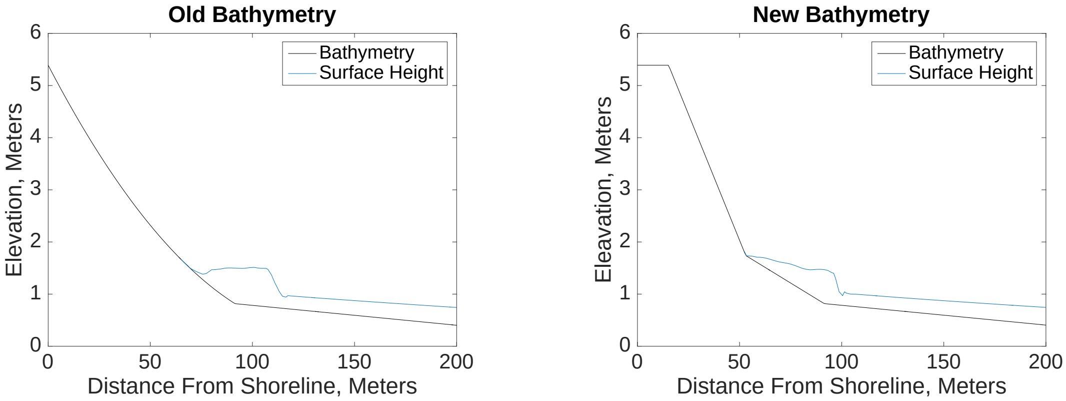


Figure 4.12: Surface height of the wave prior to reaching the shoreline for old and new bathymetry during low-tide.

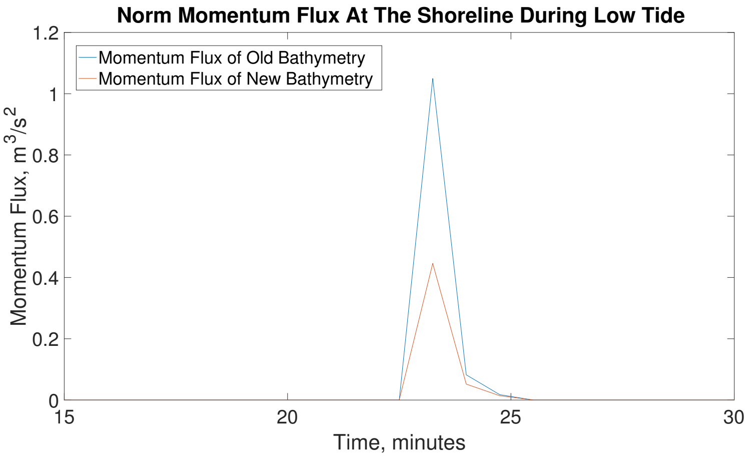


Figure 4.13: Momentum flux of different bathymetries along the shoreline during low-tide.

4.4 Error in Time Integration

Similar to the case studies in Chapter 3, we assessed the accuracy of the implicit method compared to that of the explicit method by computing the error norms. We computed the 1-Norm, 2-Norm, and the ∞-Norm for the surface height and velocity, for both low tide and high tide. With its proven accuracy, we consider explicit time integration as the fundamental truth and used it as a comparison to implicit integration. By showing little error in implicit integration we can now run the implicit time integration with a larger time-step and know we are computing quality results (up to time-truncation error).

4.4.1 Error in Time Integration for Old Bathymetry

We looked at the error norms in implicit time integration compared to explicit time integration for surface height and velocity during high tide for the old bathymetry as seen in Tables 4.1 and 4.2. The implicit method performed well with an error tolerance ranging from 10^{-2} to 10^{-1}. The norm errors for surface height and velocity during low tide are seen in Tables 4.3 and 4.4. Similarly, the error tolerance during low tide conditions ranged from 10^{-2} to 10^{-1}.

Table 4.1: Norm error for the surface height of old bathymetry during high tide from 0 to 21 minutes

	T=0	T=3	T=6	T=9	T=12	T=15	T=18	T=21
1-Norm	0	0.0241	0.0196	0.0174	0.0148	0.0115	0.0096	0.0068
2-Norm	0	0.0436	0.0395	0.0366	0.0334	0.0300	0.280	0.0229
Infinity-Norm	0	0.1021	0.0920	0.0779	0.0677	0.0527	0.0489	0.0463

Table 4.2: Error norm for the velocity of old bathymetry during high tide from 0 to 21 minutes

	T=0	T=3	T=6	T=9	T=12	T=15	T=18	T=21
1-Norm	0	0.0332	0.0273	0.0251	0.0224	0.0188	0.0160	0.0219
2-Norm	0	0.0735	0.0754	0.0735	0.0695	0.0638	0.0560	0.1162
Infinity-Norm	0	0.2647	0.2646	0.2645	0.2645	0.2645	0.2645	0.3861

Table 4.3: Error norm for the surface height of old bathymetry during low tide from 0 to 21 minutes

	T=0	T=3	T=6	T=9	T=12	T=15	T=18	T=21
1-Norm	0	0.0188	0.0149	0.0119	0.0103	0.0091	0.0070	0.0052
2-Norm	0	0.0364	0.0310	0.0254	0.0228	0.0222	0.0184	0.0154
Infinity-Norm	0	0.0678	0.0514	0.0362	0.0352	0.0325	0.0257	0.0221

Table 4.4: Error norm for the velocity of old bathymetry during low tide from 0 to 21 minutes

	T=0	T=3	T=6	T=9	T=12	T=15	T=18	T=21
1-Norm	0	0.0250	0.0199	0.0168	0.0156	0.0142	0.0118	0.0109
2-Norm	0	0.0441	0.0404	0.0357	0.0347	0.0352	0.0313	0.0360
Infinity-Norm	0	0.0910	0.0881	0.0902	0.0965	0.1092	0.0931	0.1311

4.4.2 Error in Time Integration in New Bathymetry

As we would expect, the norm errors for the new bathymetry were very similar to those for the old bathymetry during both high and low tide as seen in Tables 4.5 and 4.6. The errors ranged from 10^{-2} to 10^{-1}. During low tide, the norm errors also ranged from 10^{-2} to 10^{-1}, as seen in Tables 4.7 and 4.8.

Table 4.5: Error norm for the surface height of new bathymetry during high tide from 0 to 21 minutes

	T=0	**T=3**	**T=6**	**T=9**	**T=12**	**T=15**	**T=18**	**T=21**
1-Norm	0	0.0276	0.0299	0.0333	0.0387	0.0443	0.0515	0.0599
2-Norm	0	0.0415	0.0378	0.0338	0.0339	0.0308	0.0305	0.0294
Infinity-Norm	0	0.1080	0.0897	0.0690	0.0676	0.0539	0.0457	0.0486

Table 4.6: Error norm for the velocity of new bathymetry during high tide from 0 to 21 minutes

	T=0	**T=3**	**T=6**	**T=9**	**T=12**	**T=15**	**T=18**	**T=21**
1-Norm	0	0.0747	0.0843	0.0893	0.0905	0.0883	0.0803	0.0639
2-Norm	0	0.1880	0.02006	0.1963	0.1848	0.1664	0.1417	0.0765
Infinity-Norm	0	0.9652	1.1947	1.2504	1.2152	1.0939	0.8887	0.3247

Table 4.7: Error norm for the surface height of new bathymetry during low tide from 0 to 21 minutes

	T=0	**T=3**	**T=6**	**T=9**	**T=12**	**T=15**	**T=18**	**T=21**
1-Norm	0	0.0189	0.0144	0.0125	0.0105	0.0087	0.0069	0.0051
2-Norm	0	0.0364	0.0288	0.0263	0.0222	0.0186	0.0170	0.0145
Infinity-Norm	0	0.0627	0.0463	0.0385	0.0380	0.0282	0.0250	0.0246

Table 4.8: Error norm for the velocity of new bathymetry during low tide from 0 to 21 minutes

	T=0	**T=3**	**T=6**	**T=9**	**T=12**	**T=15**	**T=18**	**T=21**
1-Norm	0	0.0266	0.0202	0.0177	0.0161	0.0145	0.0123	0.0097
2-Norm	0	0.0461	0.0402	0.0378	0.0361	0.0346	0.0318	0.0294
Infinity-Norm	0	0.0874	0.0953	0.0911	0.1085	0.0995	0.0964	0.1106

4.5 Grid Refinement

While we demonstrated that our model produced accurate solutions, one of the major limiting factors was computational cost. Our model provided a resolution of 1.3 meters. As previously discussed, a resolution of 1.3 meters is desirable for the last 1000 meters but is not strictly necessary for the remainder of the domain. In reality, we only need a resolution of approximately 10 meters for the remainder of the domain to model a wave with a base of 20 meters.

One way to modify the resolution through the domain is with a program called GMSH, which is a finite element grid generator that has a built-in CAD engine and can take in various parameters to generate meshes used for the finite element method [42]. The program allows us to create the desired geometry for our domain and GMSH is able to produce a .msh file, which can be read by our model.

To refine our grid we decomposed the grid into two planes. One plane modeled the first 1000 meters of the domain from the shoreline (near shore plane) and the second plane represented the domain from 1000 meters to 30000 meters (deep ocean plane). See Figure 4.14.

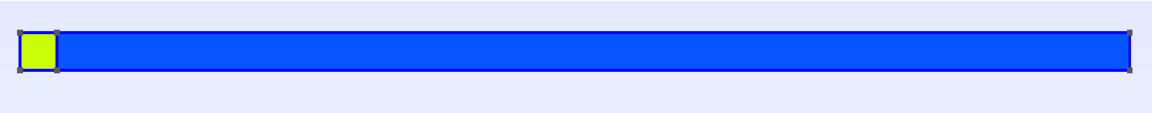


Figure 4.14: Grid divided into two planes: shoreline and deep ocean.

For the deep ocean plane, we wanted to have a resolution of every 10 meters from 1000 meters to 30000 meters. To obtain a 10 meter resolution we would need 725 elements in the x-direction of the plane and a total of 2900 points. Similar to our previous set-up, the points would be evenly spaced. Figure 4.15 represents a sample of the grid for the deep ocean plane.

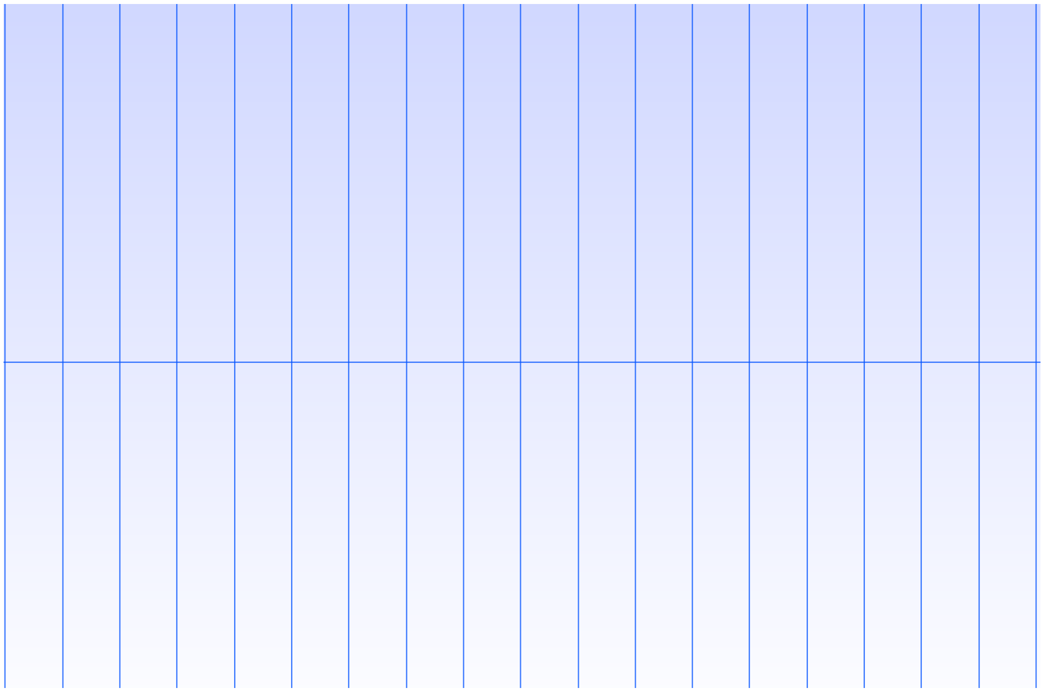


Figure 4.15: Sample of the deep ocean plane with 10 meter resolution using GMSH.

For the near shore plane, we wanted to maintain a resolution between 1-3 meters for the 100 meters closest to the shoreline. To prevent any discontinuity when meshing the near shore and deep ocean planes, we wanted our resolution for the near shore plane at 1000 meters to be close to 10 meters to match the deep ocean plane's resolution. The closer to the shoreline the more refined we made the grid. Each point had a progression of 1.009%. For example, the first point had a resolution of 1.009 meters, the second 1.018 meters, the third 1.027 meters, etc. We needed 116 elements in the x-direction for a total of 464 points to develop the near shore plane. Figure 4.16 shows the grid we used for the near shore plane.

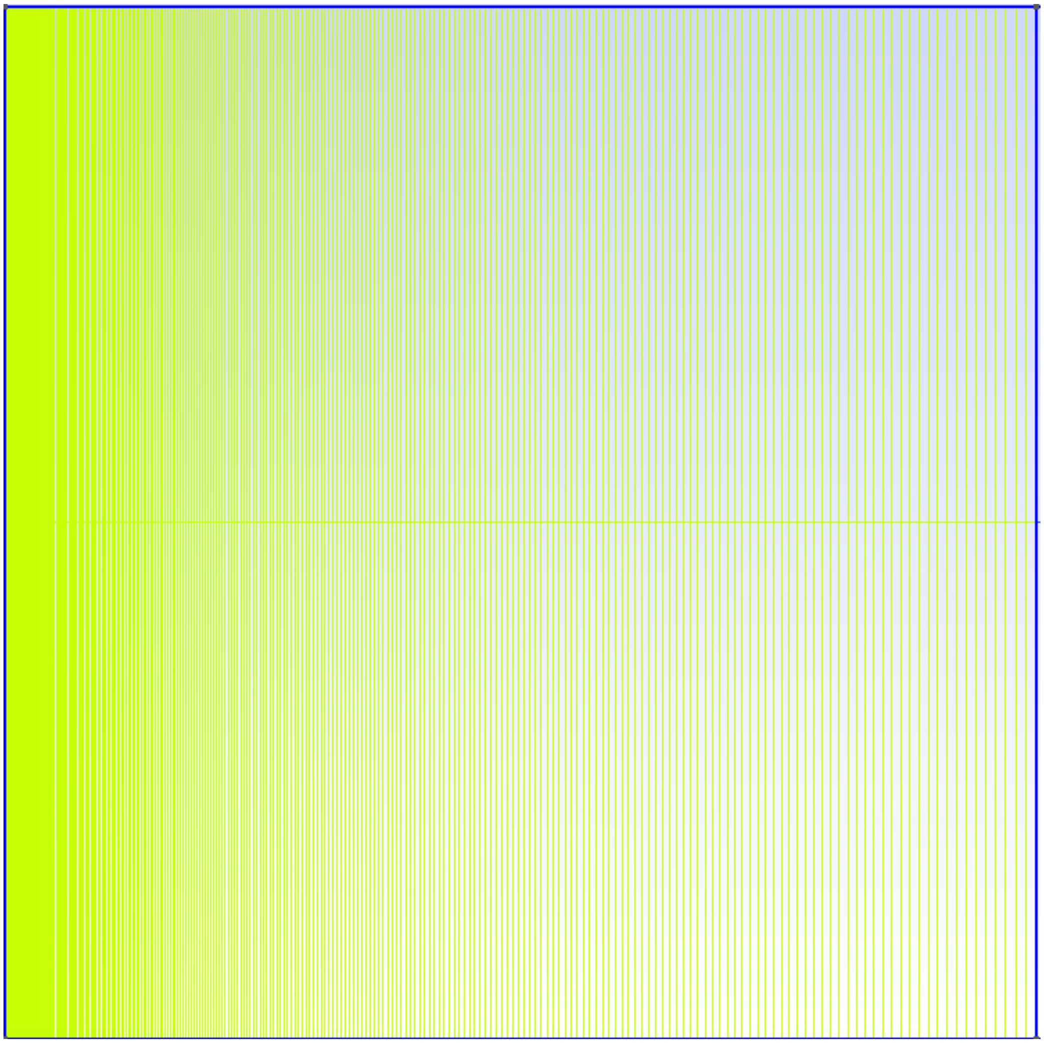


Figure 4.16: Near shore plane with resolution ranging from 1-10 meters using GMSH.

With our new grid, we needed a total of 841 elements in the x-direction for a total of 3364 points. Compared to our old configuration with 5640 elements in the x-direction for a total of 22560 points, we reduced the model by 19,196 points and were able to run the model approximately close to seven times faster than before with close to the same level of accuracy.

4.5.1 Bathymetry Discrepancy

While the model was faster, there was a small discrepancy in the new bathymetry as seen in Figure 4.15. Around the value 53.5 meters in the domain you can see that there is an inconsistency in the slope change of the GMSH grid compared to the original grid. This error is due to an issue with the grid resolution but is small enough to not impact our results in calculating momentum flux. There were no discrepancies with the GMSH grid for the old bathymetry. Since the old bathymetry is only one parabolic function it is easier to replicate compared to the new bathymetry, which has multiple changing slopes.

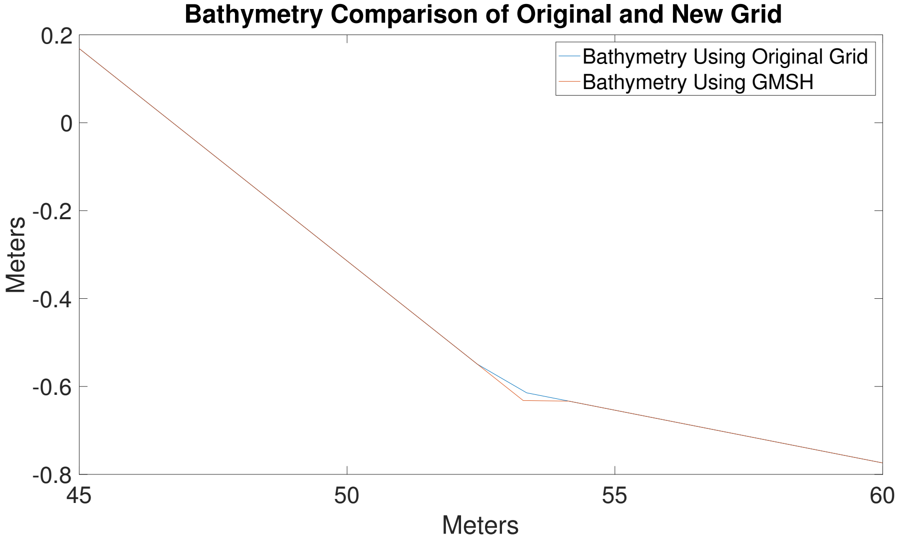


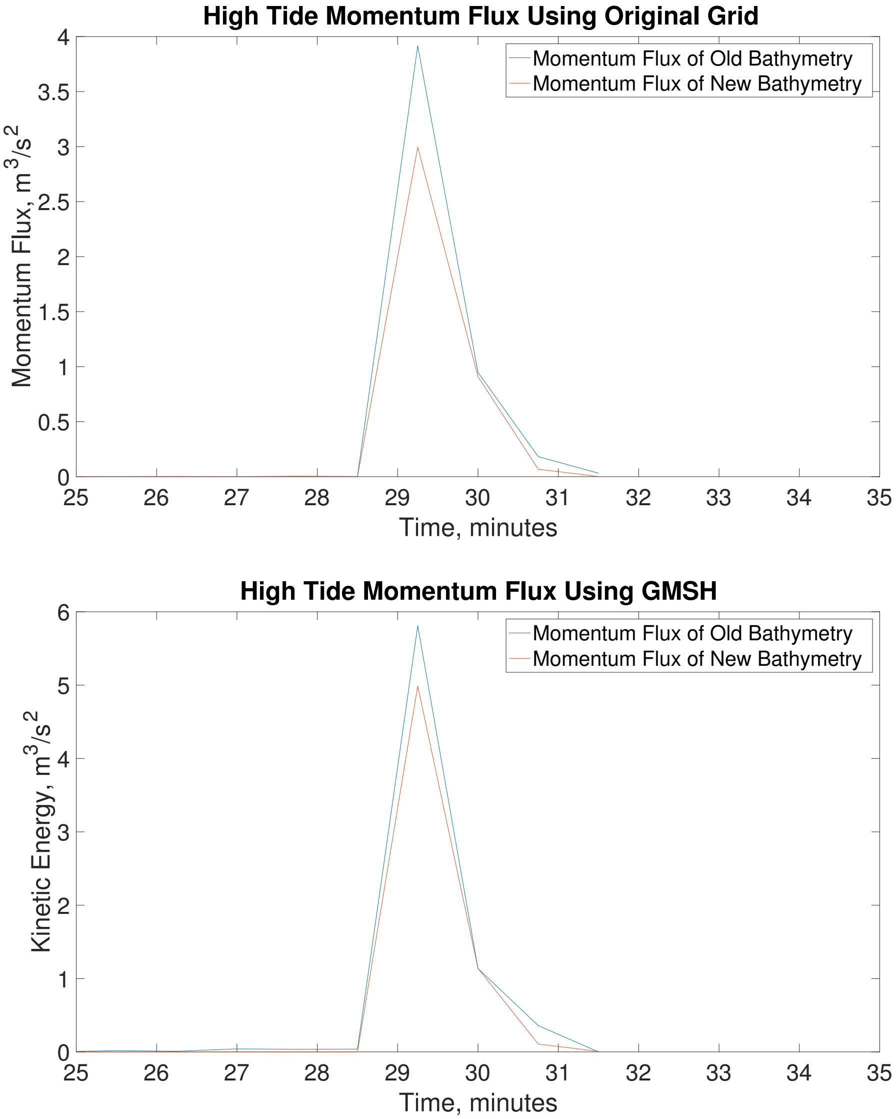
Figure 4.17: Comparison of Original grid and GMSH grid.

4.5.2 Momentum Flux Comparison

Comparing the momentum flux between the model with the original grid and the grid using GMSH you can see there are slight differences in the results. In particular, both for high tide and low tide the GMSH grid had a higher norm momentum flux value than the original grid. It is not surprising that the static adaptive grid does not match the high-grid resolution. A study by Giraldo et al. [43] shows while there is a cost savings in computational time when using an adaptive grid there is also a trade off with a decrease in accuracy.

For high tide, the peak momentum flux with the GMSH grid for the old bathymetry was $5.81\frac{m^3}{m^2}$ and the new bathymetry had a peak of $4.99\frac{m^3}{n^2}$, this is a difference of $1.91\frac{m^3}{m^2}$ and

$2.09\frac{m^3}{m^2}$ with respect to the results of the original grid. See Figure 4.18. This is over a 50% difference in values of momentum flux between the original and GMSH grid. In the future, one option to decrease the differences is to refine the grid in the proximity of the changing slope.


High Tide Momentum Flux Using Original Grid

High Tide Momentum Flux Using GMSH

Figure 4.18: Momentum flux comparison during high tide using original and GMSH grid.

For low tide, the peak momentum flux with the GMSH grid for the old bathymetry was $1.18\frac{m^3}{m^2}$ and the new bathymetry had a peak of $0.73\frac{m^3}{n^2}$, this is a difference of $0.06\frac{m^3}{m^2}$ and

$0.32\frac{m^3}{m^2}$ with respect to the results of the original grid. See Figure 4.19.

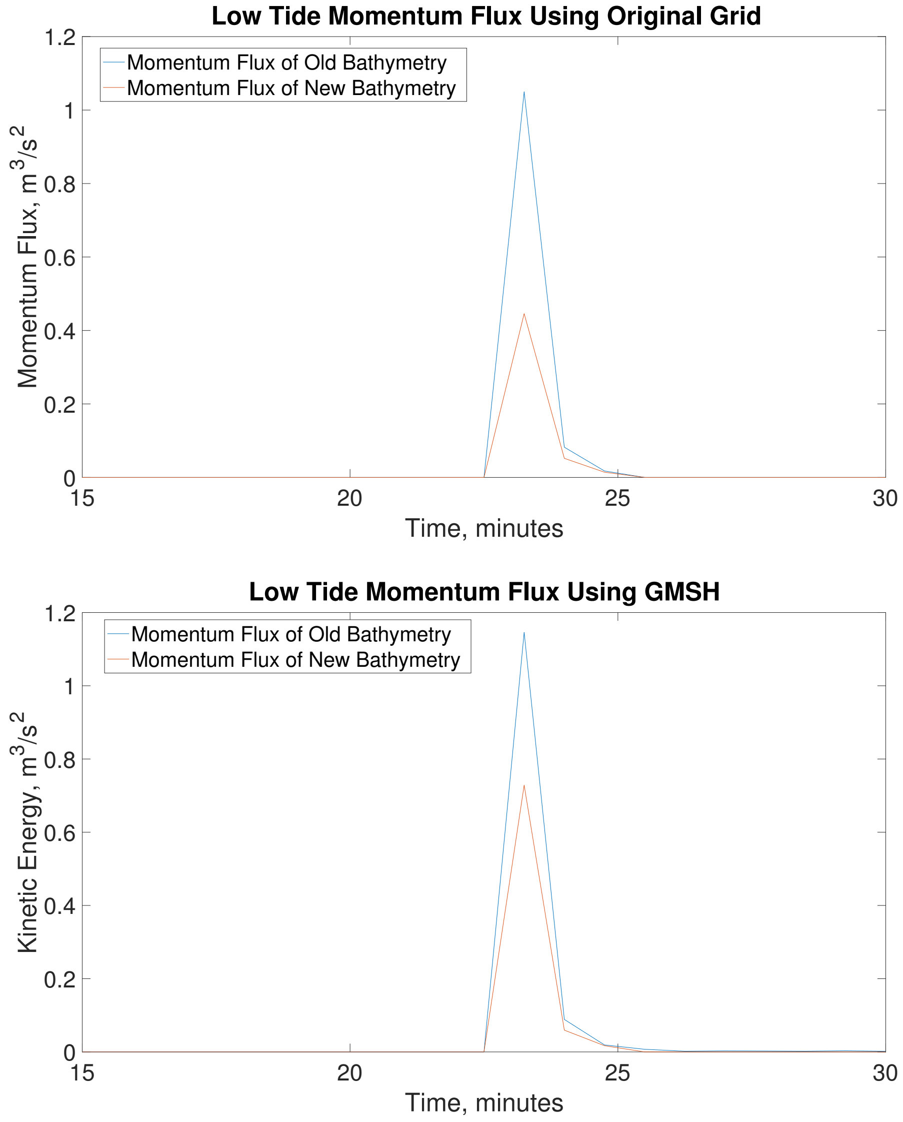


Figure 4.19: Momentum flux comparison during low tide using original and GMSH grid.

4.5.3 Quantifying Error

By going to a coarser grid (GMSH), there is resolution lost in the simulation. While there is lost information, there is a significant gain in computational speed. Using this coarser grid, we ran the model almost seven times faster than before.

To fully understand the differences in the GMSH model compared to the original model, we found the norm of the GMSH and original grids measuring bathymetry, surface height, and velocity. We then took the difference between the GMSH and original grid for bathymetry, surface height, and velocity to compare discrepancies. Since the GMSH grid was constructed differently from the original grid and it was not uniformly spaced we had to create a separate domain for comparison. We created a new grid from 0 to 1000 within 1000 evenly spaced points. Then we used a cubic spline interpolation in order to approximate our functions with the new domain. The approximation is done through dividing an interval into subintervals and then we approximate the polynomial in the subinterval [13].

The differences in bathymetry and surface height were minor compared to differences in velocity for the GMSH model. Figure 4.20 illustrates the velocity discrepancies. The velocity is higher in the GMSH model at approximately 25 minutes into the simulation. This would explain why the GMSH grid simulation had a higher momentum flux than the original.

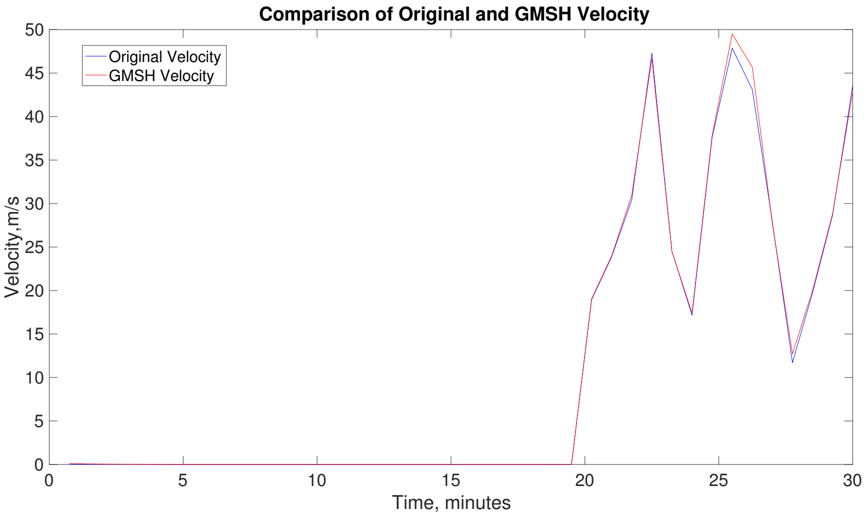


Figure 4.20: Comparison of old grid and GMSH grid.

4.5.4 Error Caused By Spurious Waves

To completely understand why there was a larger discrepancy for the water velocity compared to the minimal differences seen in the bathymetry and surface height we ran a full time lapse video overlapping the two grids velocity output frames for both the 1000 meter domain and the 30,000 meter domain. Running a full time lapse video we noticed that the velocity caused by the spurious wave was reflecting on the right boundary wall and interacting with our simulation.[4]

In order to run our model we had to have an initial wave in the deep ocean. The initial wave broke into two waves. See Figure 4.21. One wave was an authentic wave and the other wave is a spurious wave. The authentic wave traveled towards the shoreline while the spurious wave traveled to the deep shore where it hit the right boundary of the model. After hitting the boundary wall the wave reflected and traveled back towards the shoreline.

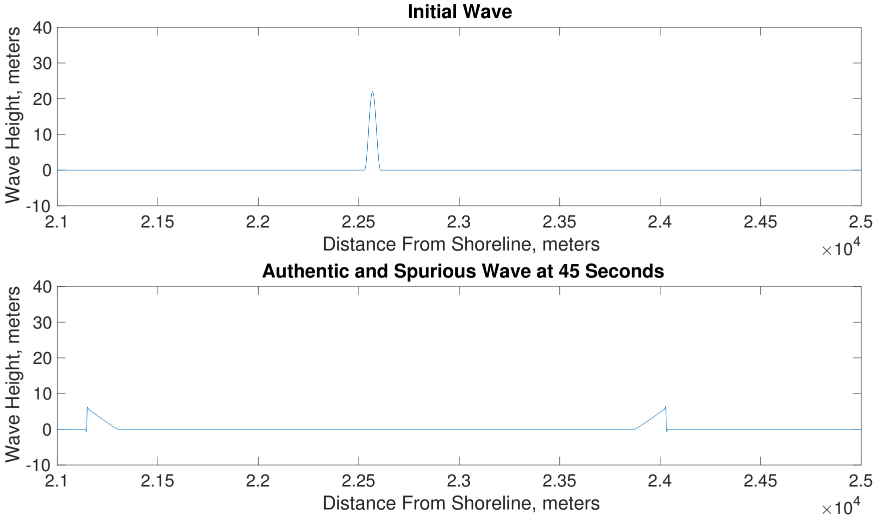


Figure 4.21: Authentic wave and spurious wave.

There was always a concern about the spurious wave's artificial impact on the two bathymetries. To address the issue we ran a time lapse video of the surface height of the model. In our time lapse video the surface height of the false wave did not reach our domain of concern till after the authentic wave reached the shoreline. What was not initially looked at was the impact of the water velocity from the artificial wave.

The spurious wave impacted the velocity more on the GMSH grid than the old grid. Since the GMSH grid had a significantly coarser resolution in the deep sea plane there was an increase of numerical dispersion error across the grid. The numerical dispersion error led to a phase error of increase magnitude as the spurious wave traveled through the domain [44]. This phase error also aided in the discrepancies in the comparison of momentum flux.

---

[4] A spurious wave is a wave that should in principle travel out to the open ocean but due to our boundary conditions is reflected back into the model.

4.6 Refined Boundary Conditions

There are a couple of possibilities to overcome the velocity discrepancies with the spurious wave. The first option is to increase the domain size. An increase in the domain size would allow for a longer travel time of the spurious wave and delay the wave from reflecting and interacting with the different bathymetries. While this method would work, the computational cost would significantly increase and be counterproductive in developing a model that is computationally efficient.

Another option would be to establish a sponge layer non-reflecting boundary condition (NRBC) on the right wall of the model. A NRBC is when an artificial boundary, B, is created, which truncates the original domain, Ω, and also creates a residual domain, D. When the spurious wave enters the artificial boundary, the wave is eliminated and does not reflect back into the original domain [45].

We would add a NRBC to our code and then solve the time dependent problem numerically in the original domain such that given our original semi-discrete equation (Eq 2.23),

$$\frac{d\mathbf{q}}{dt} = S(\mathbf{q}), \tag{4.6}$$

we solved the equation explicitly as

$$\mathbf{q}^{n+1} = \mathbf{q}^{n} + \Delta t S(\mathbf{q}^{n}), \tag{4.7}$$

where $\mathbf{q}^{n+1} = \mathbf{q}^{explicit}$.

Then with an NRBC in place,

$$\mathbf{q}^{n+1} = \alpha \mathbf{q}^{explicit} + \beta \mathbf{q}^{boundary}, \tag{4.8}$$

where for the following conditions,

$\alpha = 1$, then $\beta = 1 - \alpha$, or

$\alpha = 0$, then $\beta = 1$.

While we did not use a NRBC, in future simulations we will apply these new conditions.

4.7 Chapter Summary

Based on our model inputs, our predictions that the new bathymetry would have less momentum flux coincided with the results of our simulation. There was less momentum flux at the shoreline for the new bathymetry compared to the old bathymetry. Therefore, the beach restoration was an effective project in reducing energy along the shoreline during storm surges. This reduction in energy means there would be less destruction from the storm surge with the new bathymetry.

While we had to use a higher grid resolution, which increased the computational time, we were able to provide accurate solutions both with implicit and explicit time integrators. We ran our model on the Naval Postgraduate School's supercomputer, also known as the "Hamming" computer (named after world-renowned mathematician Dr. Richard Hamming), in order to meet the high random access memory (RAM) demand. Since we were using a large grid, Hamming's large RAM proved to be very beneficial to run our model. Hamming is a Sun Microsystems High-Performance Cluster that has a processing power of 10.736 teraflops (10.736 trillion floating-point operations per second) [46]. To run each simulation, it took approximately 80 hours of processing time on Hamming. In the future, if we parallelize our code, the simulations will run faster since we will have the flexibility to use more than one core of Hamming.

After running our simulation at high resolution, we then modified our grid to varying resolutions depending on the location of the domain. The grid modification allowed for a faster computational time. While the model ran significantly faster we did sacrifice some resolution due to a coarser grid. We also learned the impacts a spurious wave has on water velocity and momentum flux.

CHAPTER 5:
Conclusion

Our study used the DG method to solve the SWE on quadrilaterals. Prior to running our simulation of two bathymetries in La Push, WA, we verified our method with three test cases using both explicit and implicit time integrators. The verification of our method was essential prior to simulating a storm surge. During our verification, we also determined the error for the implicit method compared to the explicit method. We calculated the 1-norm, 2-norm, and ∞-norm and found that our error ranged from 10^{-2} to 10^{-1} in magnitude, which was within our desired tolerance. Demonstrating the effectiveness of the implicit time integrator gave us the flexibility to use large time-steps for future work. After our test verification was completed, we then simulated a beach restoration project in La Push, Washington. Based on field data, we simulated an 11-meter wave moving across 30,000 meters and breaking on the old and new bathymetry. The momentum fluxes were calculated and compared to determine if the new bathymetry was more resistant to a storm surge.

Throughout this study, we focused on the most optimal resolution with computational efficiency. After we initially ran our model, we realized there were discrepancies and therefore increased the resolution from 7520 points to 22560 points. High resolution of the bathymetry was important because if we had poor resolution then we would have inaccurate results for computing momentum flux.

After running the model at a high resolution, we modified the grid resolution to vary throughout the domain with a focus on high resolution closer to the shoreline where we were more concerned. The grid modification allowed us to save in computational time by a factor of 6.7. While we saved in computational time, we did sacrifice some accuracy due to the coarser grid. We also saw the impacts a spurious wave has on velocity and momentum flux for a coarser grid.

Our results showed that the storm surge had less momentum flux along the shoreline for the new bathymetry compared to the old bathymetry for both high tide and low tide storm surges. Therefore, the new bathymetry was more effective against the destruction of storm surges. In particular, the new bathymetry was more effective during a low-tide storm surge *compared to a high-tide storm surge.*

5.1 Limitations

Prior to this study, the code that we used had not been verified against similar models. Verification of the code was a limiting factor due to the time that was required prior to being able to run our simulation. In addition, without the ability to physically replicate our model with the use of wave tanks in a laboratory, we had to use previous numerical simulations for comparison. In particular, we used the Balzano test cases for comparison. With a domain of 30,000 meters and a wave height of 11-meters, it was not feasible to run a full-scale laboratory experiment of a storm surge of this magnitude; however, it would still have been desirable to have a scaled-down physical model of the simulated storm surge.

We were limited by available input data. Given the remote location of La Push, Washington we were constrained with limited data and had to make some assumptions for the simulations. Some of these assumptions included having a uniform bottom friction constant of 0.02 and also assuming the bathymetry after 100 meters from the shoreline was linear till reaching Buoy Station 46041. In addition, our high-tide and low-tide conditions came from the U.S. Harbors and covered the general region of the Olympic Peninsula and was not specific to La Push. While we were limited in data input, we were able to use multiple resources to help substantiate our assumptions.

Lastly, we also were constantly facing the issue of computational efficiency versus resolution. When you are primarily interested in the quality of the solution in the last 100 meters of a large domain, you need to determine an appropriate number of elements. Determining the number of elements, allows us to optimize the balance between computational efficiency and resolution. For our model, we used a fourth order polynomial with two elements in the y-direction and 5640 elements in the x-direction for a total of 22560 points. We could have continued to add elements in the x-direction, but computation time limited our model to 5640 elements.

5.2 Future Work

Adding a NRBC to our model will be desirable to eliminate the spurious wave. With an addition of a NRBC, we would run both bathymetries with the original grid and GMSH grid and there would be less of a difference in velocities and momentum flux.

Our simulation only took into account one wave, but in reality, a storm surge would have multiple waves of different amplitudes. Since we have an effective model to simulate a single wave surge, the next logical step would incorporate multiple waves into the model. Multiple waves would add an extra layer of complexity by incorporating the energy caused by waves interacting with each other.

In addition, all our simulations were in 2-D but for a 1-D initial condition. In the future, we could adapt the simulations to 3-D. Since SWE are, at the most, 2-D by definition, to run a full 3-D ocean simulation you need to switch to a Navier-Stokes solver. Moreover the combination of 3-D with adaptive mesh refinement (AMR) would provide a computationally efficient model that could focus on the changes in bathymetry of the last 100 meters in our domain [43]. AMR allows models to have high resolution in specific areas that are critical to the domain.

Another area for future consideration is the movement of sediment due to erosion from storm surges. The ability for a beach restoration to be sustainable and resistant to erosion is just as critical as being able to mitigate the momentum flux produced from a wave. Modeling sediment movement has been an extremely difficult problem for researchers studying the coast and navigation channels due to the complexity of the problem. It is of particular interest to USACE because sediment transport impacts the navigation of channels and also erosion due to sediment transport can impact homes and other infrastructures built along shorelines. One equation that has been widely used for sediment transport is the advection-diffusion equation. The advection-diffusion equation is a PDE used to numerically represent sediment movement. In the future, the DG method could be used to solve this PDE [47].

THIS PAGE INTENTIONALLY LEFT BLANK

List of References

[1] National Oceanic and Atmospheric Administration (NOAA). Storm surge overview. [Online]. Available: http://www.nhc.noaa.gov/surge/

[2] C. Vreugdenhil, *Numerical Methods For Shallow-Water Flow* (Water Science and Technology Library), L. S. U. V.P. Singh, Ed. New York City, New York: Springer, 1994, vol. 13.

[3] S. Marras, J. Kelly, M. Moragues, A. Meller, M. Kopera, M. Vazquez, F.X. Giraldo, G. Houzeaux, and O. Jorba, "A review of elements-based Galerkin methods for numerical weather prediction. finite elements, spectral elements, and discontinuous Galerkin," *Archives of Computational Methods in Engineering*, 2015.

[4] E. Kubatko, "Development, implementation, and verification of HP discontinuous Galerkin models for shallow water hydrodynamics and transport," Ph.D. dissertation, Notre Dame, 2005.

[5] B. Bonev, "Discontinuous Galerkin methods for shallow water equations," Master's thesis, Department of Computational Mathematics and Simulation Science, EPFL Lausanne, Lausanne, Switzerland, November 2015.

[6] C. Dawson, J. J. Westerink, J. C. Feyen, and D. Pothina, "Continuous, discontinuous and coupled discontinuous-continuous Galerkin finite element methods for the shallow water equations," *International Journal for Numerical Methods in Fluids*, vol. 52, pp. 63–88, February 2006.

[7] D. Alevras, "Simulating tsunamis in the Indian Ocean with real bathymetry by using a high-order triangular discontinuous Galerkin oceanic shallow water model," Master's thesis, Naval Postgraduate School, March 2009.

[8] L. N. Trefethen and I. David Bau, *Numerical Linear Algebra*. Philadelphia, Pennsylvania: Society for Industrial and Applied Mathematics SIAM, 1997.

[9] M. Taylor, J. Tribbia, and M. Iskandrani, "The spectral element method for the shallow water equations on the sphere," *J. Comp. Phys*, vol. 130, pp. 92–108, 1997.

[10] R. D. Nair, S. J. Thomas, and R. D. Loft, "A discontinuous Galerkin global shallow water model," *Scientific Computing Division, National Center for Atmospheric Resarch*, 2004.

[11] D. Wiraset, E. Kubatko, C. Michoski, S. Tanaka, J. Westerink, and C. Dawson, "Discontinuous Galerkin methods with nodal and hybrid modal/nodal triangular, quadrilateral, and polygonal elements for nonlinear shallow water flow," *Computer Methods in Mechanics and Engineering*, vol. 270, pp. 113–149, 2013.

[12] B. Cockburn, J. Gopalakrishnan, and H. Wang, "Locally conservative fluxes for the continuous Galerkin method," *Society for Industrial and Applied Mathematics*, 2007.

[13] R. Burden, *Numerical Analysis*, 9th ed. Pacific Grove, California: Brooks Cole Pub., 2011.

[14] F. X. Giraldo, "Element-based Galerkin methods on tensor-product bases," Lecture Notes (unpublished).

[15] U. A. C. of Engineers, "La Push coastal emergency: Direct assistance to Quileute Nation Clallam County, Washington," U.S. Army Corps of Engineers, Tech. Rep., January 2012.

[16] I. Evans-Hamilton, "FY 2010 field data collection program at the Quillayute navigation project, La Push, WA: Data report," Evans-Hamilton, INC., Tech. Rep., June 2010.

[17] C. Villaret, J.-M. Hervouet, R. Kopmann, U. Merkel, and A. Davies, "Morphodynamic modeling using the telemac finite-element system," *Computers and Geosciences*, 2013.

[18] F. X. Giraldo, "Introduction to DG methods," in *Gene Golub SIAM Summer School 2012*, 2012.

[19] F. X. Giraldo and M. Restelli, "A study of spectral element and discontinous Galerkin methods for the Navier-Stokes equations in nonhydrostatic mesoscale atmospheric modeling: Equation sets and test cases," *Journal of Computational Physics*, vol. 227, pp. 3849–3877, 2007.

[20] F. X. Giraldo, "The Lagrange-Galerkin spectral element method on unstructured quadrilateral grids," *J. Comp. Phys*, vol. 147, pp. 114–146, 1998.

[21] W. Schiesser, *The Numerical Method of Lines. Integration of Partial Differential Equations*, 1st ed. Cambridge, Massachusetts: Academy Press, Inc., 1991.

[22] F. X. Giraldo, "Implicit-explicit time-integration for a unified continuous and discontinuous Galerkin discretization of the Euler equations," unpublished.

[23] J. Escobar-Vargas, P. Diamessis, and F.X. Giraldo, "High-order discontinuous element-based schemes for the inviscid shallow water equations: Spectral multidomain penalty and discontinous Galerkin methods," *Applied Mathematics and Computation*, vol. 218, no. 9, pp. 4825–4846, January 2012.

[24] Z. Kowalik and T. Murty, “Numerical modeling of ocean dynamics,” *World Scientific*, 1993.

[25] R. Reid and B. Whitaker, “Wind-drive flow of water influenced by a canopy,” *J. Waterways, Harbors Coastal Engineering Division*, pp. 61–77, 1976.

[26] A. Balzano, “Evaluation of methods for numerical simulation of wetting and drying in shallow water flow models,” *Coastal Engineering*, vol. 34, pp. 83–107, 1998.

[27] Y. Xing, X. Zhang, and C.-W. Shu, “Positivity-preserving high order well balanced discontinuous Galerkin methods for the shallow water equations,” *Advances in Water Resources*, vol. 33, pp. 1476–1493, 2010.

[28] F. X. Giraldo, “Time-integrators,” unpublished.

[29] D. Knoll and D. Keyes, “Jacobian-free Newton-Krylov methods: A survey of approaches and applications,” *Journal of Computational Physics*, 2004.

[30] P. L. Butzer and R. Weis, “On the lax equivalence theorem equipped with orders,” *Journal of Approximation Theory*, 1977.

[31] O. Gourgue, R. Comblen, J. Lambrechts, and E. D. Tuomas Karna, Vincent Legat, “A flux-limiting wetting-drying method for finite-element shallow-water models, with application to the Scheldt Estuary,” *Advances in Water Resources*, 2009.

[32] A. D. R. Corps of Engineers and Operations, “Public Law 84-99 rehabilitation assistance for non-federal flood control projects,” Department of the Army, Tech. Rep., 2009.

[33] U.S. Army Corps of Engineers, “Environmental assessment Fiscal Years 2009-2014 Quillayute River navigation channel maintenance dredging,” U.S. Army Corps of Engineers, Tech. Rep., 2009.

[34] R. S. Anderson and S. P. Anderson, *Geomorphology The Mechanics and Chemistry of Landscapes*. Cambridge, England: Cambridge University Press, 2010.

[35] N. Oceanic and A. A. (NOAA). (2015, September). National Data Buoy Center. [Online]. Available: http://www.ndbc.noaa.gov/station_history.php?station=46041

[36] G. J. Arcement, “Guide for selecting Manning’s roughness coefficients for natural channels and flood plains,” United States Geological Survey, Tech. Rep., 1989.

[37] “Station 46041. 47.353 N 124.731 W . Google Earth. October 2013. February 26, 2016.”

[38] I. White, D. Mottershead, and S. Harrison, *Environmental Systems An Introductory Text*. Cheltenham, Great Britain: Stanley Thornes Ltd, 1998.

[39] U. Harbors. (2015). Washington tide charts US harbors. [Online]. Available: http://wa.usharbors.com/monthly-tides/Washington-Columbia

[40] H. Park, D. t. Cox, P. J. Lynett, D. M. Wiebe, and S. Shin, "Tsunami inundation modeling in constructed environments: A physical and numerical comparison of free-surface elevation, velocity, and momentum flux," *Coastal Engineering*, vol. 79, pp. 9–21, 2013.

[41] R. Dean and R. Dalrymple, *Water Wave Mechanics For Engineers and Scientists*. Singapore: World Scientific, 1991, vol. 2.

[42] C. Geuzaine and J.-F. Remacle, "GMSH: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities." *International Journal for Numerical Methods in Engineering*, vol. 79, no. 11, pp. 1309–1331, 2009.

[43] M. A. Kopera and F. X. Giraldo, "Analysis of adaptive mesh refinement for IMEX discontinuous Galerkin solutions of the compressible Euler equations with applications to atmospheric simulations," *Journal of Computational Physics*, vol. 271, pp. 92–117, 2014.

[44] J. Nehrbass, J. Jevtic, and R. Lee, "Reducing the phase error for finite-difference methods without incrasing the order," *IEEE Transactions of Antennas and Propagation*, vol. 46, no. 8, pp. 1194–1201, August 1998.

[45] D. Givoli, "High-order local non-reflecting boundary conditions: A review," *Wave Motion*, vol. 39, no. 4, pp. 319–326, April 2004.

[46] Hamming Supercomputer Boosts NPS Research. [Online]. Available: http://www.nps.edu/Technology/Documents/SUPERCOMPUTER

[47] P. Y. Julien, *Erosion and Sedimentation*. Cambridge, England: Cambridge University Press, 2010.

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California